Convivial Decay:
Entangled Lifetimes in a Geriatric Infrastructure

Marisa Leavitt Cohn
IT University of Copenhagen
Rued Langgards Vej 7
2300 København S
marisa@itu.dk

ABSTRACT
This paper discusses the empirical case of an aging and obsolescent infrastructure supporting a space science mission that is currently approaching a known end. Such a case contributes to our understanding of the degrading path at the end-of-life of an infrastructure. During this later stage in the life of infrastructure we can observe common issues associated with aging infrastructures – hardware's material decay, programming languages and software tools reaching end of support, obsolete managerial methodologies, etc. Such a case of infrastructural decay reveals how work of infrastructure maintenance may reach the limits of repair and shift from repair-as-sustaining into a mode of repair-into-decay, actively working towards the end-of-life. What this reveals is that, rather than infrastructural decay being a natural by-product of time's passing, there is active work that goes into producing a convivial decay in which the multiple temporalities of aging and decay are brought into alignment through negotiation of what aging means, its impacts on different forms of work, and even what counts as old and new.

Author Keywords
materiality; temporalities; life cycles; lifetimes; repair; sustainability; longevity; infrastructure studies; maintenance

ACM Classification Keywords
H5.m Information interfaces and presentation (HCI): Miscellaneous.

General Terms
Human Factors, Management, Theory.

INTRODUCTION
Recent work in CSCW has turned attention to studies of technology that go beyond early phases of technology design and adoption to include "downstream" contexts of technology repair and maintenance [11–13,17,21,24]. Along with work on departures and abandonment of technologies [1,10], as well as calls within sustainable HCI for design approaches that will take disposal into account [3,19], these works aim to redress a stated bias in the literature towards accounts of novel technologies in early stages of adoption. They also align with calls within infrastructure studies to examine the full "biography of artifacts" by studying infrastructures longitudinally through the various stages of their life cycle [15,21].

As Jackson et al point out, this is not an entirely new focus for CSCW research, which has historically attended to invisible forms of labor that go into creating and sustaining technologies, from studies of articulation work [27], to appropriation [6], to work-arounds [5], and break-downs [20,29]. What all this work acknowledges is the multiplicity of working relations to technology that emerge over the lifetime of a technological object as it shifts from sites of design and adoption, into repair, reuse, appropriation, disposal, and abandonment. Given this multiplicity, there is value in gaining an empirical understanding of cases of technologies at a variety of stages in the technological life cycle.

This paper focuses on the stage of life when a technological infrastructure is nearing its end-of-life. While others have examined abandoned databases [10] users abandoning social media technologies [1], and the disposability of technologies [3] there is yet to be a study of the empirical case of the end-of-life of an infrastructure. Such a case, I will argue, not only fleshes out the particular empirics of this stage of life, but also troubles the assumed temporalities of technological life cycles.

Based on an empirical study of a degrading space science mission infrastructure, I find that infrastructural decay, rather than being a natural by-product of innovation, is actively negotiated alongside an emerging appreciation for the relative agedness of different parts of the infrastructure. This knowledge of agedness emerges through reflection on sociotechnical change not as rupture, but as drift over the duration of infrastructure's multiple lifetimes. Agedness is not only known, but is actively performed and advocated for, throughout this negotiation. As infrastructural decay is negotiated in terms of what should decay and how decay, as a process, should unfold, a particular form of repair-into-

CSCW '16, February 27-March 02, 2016, San Francisco, CA, USA

ACM 978-1-4503-3592-8/16/02 $15.00
DOI: http://dx.doi.org/10.1145.2818048.2819993

decay emerges which I call convivial decay[1]. Convivial decay, I argue, forms a practice of alignment work [14], translating across multiple lifetimes, while working actively towards the end.

RELATED WORK

Much research on later stages in the life cycle of technology focuses on the active work of repairing, maintaining, or appropriating technology. This work often implicitly restores to technological practices the privileged status of design and innovation by revealing design-in-use or design-in-appropriation [17]. Similarly, ethnographic work on repair demonstrates the ingenuity and creativity [17] in our "diverse… capacity for repair: the ability to make broken and breaking systems work" [13].

This work on repair takes a fundamental epistemological stance that it takes active work to make "*any* sociotechnical arrangement… to work and persist through time" [13]. While I align with this perspective that repair involves ongoing active work to sustain technology, I also note that it places emphasis on repair as a form of renewal. As Graham and Thrift state: "The world is involved in a continuous dying that can only be fended off by constant repair and maintenance" [8]. Change and flux are constant and therefore repair and maintenance are often examined as sites of ongoing renewal, keeping things going, to "sustain them for at least a little while longer" [13].

Sims and Henke [25] have suggested a distinction between repair-as-maintenance (keeping the status quo) and repair-as-transformation (e.g. retrofitting) (cited in [26]). As Sims points out, both degradation and obsolescence are forms of "slippage" between what a system was designed to do and what it is able to do, either through aging of components that no longer perform as intended or because of changing demands, standards, or perceptions of performance [26]. Infrastructural repair in the form of upgrades, updates, work-arounds, etc. "are thoroughly sociotechnical activities" in that they "involve restoring both technology and social order" [26]. Determining both how a system "is" and what it "ought" to be like are interpretive processes that call on different kinds of technical and social organizational expertise [26].

This relates to the notion of "repair" derived from ethnomethodological studies of "conversational repair" [7] which have been applied as well to understanding technology repair-in-use [29]. In this notion of repair, it is the relationship between what a technological system ought to do (or is thought to do) and what it does that is repaired. This can be achieved both through changes to the technology itself or through changes to expectations.

However, in a case where the end-of-life of a system is known and impending, there is a shift in thinking about repair. An infrastructure entering old age and approaching death may have reached a time beyond repair, where, rather than fending off death, repair is part of working towards its end. Rather than a form of repair to fend off death or restore the status quo, I observed, in the case that follows, a shift towards repair-into-decay. This involves an acceptance of decay, through an understanding that emerges that the infrastructure is not what it used to be: a recognition of a need to slow down, let go of expectations, and make cuts to functionality. As in all cases of repair, this repair is still oriented towards extending the life of the system so that it does not end prematurely. But there is an increasing acceptance that repairs are made in an effort to decay "gracefully" rather than break catastrophically.

What such a case offers, then, is an example of what Jackson et al has called a "recessionary informatics" –in which we might imagine a "fundamentally contracting technological world… [of] breakdown, withdrawal, and decline" [13] or in which we might attend as much to the importance of "disconnection and disassembly" as to "connection and assembly" in our technological worlds [8]. A system approaching the end of its life allows us to observe this downgrade path, examining repair's ability to keep things going, but also the limits of repair.

It is not surprising that there is little research on dying infrastructures, since we might assume that something that is coming to an end has little insight to offer about current or future practice. Yet this assumption is premised on a progressive account in which dying systems are those that have ceased to be relevant. It is also an assumption premised upon a future that moves towards expansion and greater capability, as has been challenged in [30], where working with more limited capabilities would have little value.

This may arise from an implicit orientation within CSCW towards understanding sociotechnical change as an evolutionary process. From this perspective, technologies that decline and go away are those that did not succeed because they were unsustainable or unfit. This is a view in which the past is irrelevant, except as a form of historical narrative to understand how we came to be where we are. From a computational perspective, we simply outlive and move beyond forms of computational work that are no longer relevant (e.g. who wants to read/write in hex code?!). Obsolescence, degradation, and decay, according to this view, would be merely a natural by-product of innovation and time's passing and would not require explanation on their own terms.

As this paper will argue, this is far from the case. Infrastructural decay, as any form of decay, is not simply a

[1] I appreciate one of the reviewers of this work for pointing me to the work of Ivan Illich who has written about conviviality of technological tools. While the concept has relevance to this work, I have drawn primarily on the work of Donna Haraway and her concept of livable worlds in conceptualizing conviviality.

backdrop against which progress takes place. A dying system is not simply a forked evolutionary path that has reached a dead end. The dying of an infrastructure is an active pursuit and indeed what lives or dies in infrastructure is always open to negotiation. If there is something we might deem infrastructural decay it is composed of multiple lifetimes of different parts of the system – hardware, software, code, organizational processes, programming languages, institutions, careers – all of which are entangled and are aging or obsolescing at different rates. Infrastructural decay is not a natural or essential process of falling away, but rather an outcome or achievement made through active moves and cuts within these entangled lifetimes. In the case that I describe, what emerges is an understanding of "convivial" decay in which the relations across these multiple lifetimes are negotiated so that the infrastructure can age gracefully.

METHODS

This paper presents findings from nine months' ethnographic fieldwork examining the work of engineers to maintain and operate a large-scale, multi-decade technological infrastructure, built to support a space science mission (hereafter referred to as the Mission) at a major laboratory in southern California (hereafter referred to as the Lab). This infrastructure supports the science and engineering teams at the Mission in their work to meaningfully operate and command a spacecraft that is currently in orbit around Saturn for the purpose of scientific data collection.

Designed and built in the 1980s and 90s, and launched in 1997, the spacecraft reached Saturn in 2004 and has conducted scientific data collection across twelve different scientific instruments. In addition to the spacecraft itself, which has "on board" software for flight and instrument operations, the infrastructure includes the "ground system" comprising software tools for designing and coordinating science observations and for sending commands to the spacecraft.

The Mission organization includes scientific teams distributed across the US and Europe who participate in selecting, designing, and implementing scientific observations. The team at the Lab comprises both disciplinary scientists conducting Saturn science as well as the engineers who operate and command the spacecraft as a whole, as well as several of its instruments.

My initial questions that motivated the fieldwork were focused on how software tools mediate distributed collaborative work and translation across disciplinary boundaries. This motivation informed my methods, which included interviews with engineers working in different disciplines from navigation, spacecraft operations, science and mission planning, and software development; shadowing of work practices; attending formal and ad hoc meetings; and walk-throughs of software tools.

Data collection included semi-structured interviews with 30 key informants from across different teams at the Mission (navigation, engineering, science planning, e.g.) as well as participant observation across these different teams. These observations included dedicated time (3-4 weeks) with each team as well as observations that followed a particular work product (a sequence of science observations) through the Mission workflow process. I was present for 3-4 days a week at the organization and sat in on approximately 200 formal and informal meetings.

As I realized the salience of the long-livedness and agedness of the infrastructure, I began to incorporate interview methods drawn from oral histories methods to discover more about how the infrastructure and the mission's work had evolved over the years, particularly for engineers who had spent many years on the mission. I returned for follow up interviews with key informants, asking them about their work at the Mission within the context of their career and how the Mission and its infrastructure had evolved over the duration of their work there.

A GERIATRIC INFRASTRUCTURE

At the time of my fieldwork, the Mission was one of the only outer planet missions in operations at the Lab, considered a "flagship" mission both in terms of the size of its distributed scientific community and in terms of budget and personnel at the Lab. It was also immediately apparent upon arriving for my fieldwork that the Mission is one of the oldest on the Lab that is still in operations. It was often referred to as a "dinosaur" or a "well-oiled machine" – terms that denoted both frustration and pride about the fact that the mission has its own way of doing things. The software team was continually fending off pressure to upgrade to newer centrally developed Lab-wide systems or failing in their attempts to do so. Some engineers who have spent a decade or more on the Mission are accustomed to computing work that would be difficult to find still practiced elsewhere, such as manually reading hex code, or coding in tcl.tk.

The Mission is in fact comprised of several phases each of which comprised its own mission plan and bid for funding to NASA. It entered its third phase in 2010 after being granted a final extension of the mission. It was during the transition from the middle to final phase of the mission in 2009-2010 that I conducted my fieldwork. The final phase lasting from 2010-2017, while seven years long, was approved for a reduced budget and so this transition involved cuts to funding and personnel (most of whom moved to other missions at the Lab). This transition to the final phase came with a decline in resources and personnel, as NASA expects such a "well-oiled" machine to require less funding to do what it already does well.

The start of the final phase also meant that the engineers were beginning to plan for the end. Multiple proposals for

the final phase of the mission were put forth, depending on the funding received, but all included a "spiral of death" in the final year of the mission, when the spacecraft would fly in tighter and tighter orbits, eking out its last drops of its finite "consumables" such as fuel, to reach a spectacular finish, flying between the planet and its rings before plummeting into the planet to burn up in its atmosphere. This death spiral allows the team to continue science up until its last moments until either fuel runs out or communication is lost, imparting little boosts to maintain the spacecraft's orbit as long as possible. The final "naturally decaying" orbit also ensures the proper "disposal" of the spacecraft - if communication is lost, the spacecraft will not escape the gravitational pull of Saturn, but will "go ballistic" and plummet to its death.

During this time, the spacecraft also began to show various signs of decay. One of its "reaction wheels" was showing signs of drag; one of its instruments was occasionally shorting out. And on the software side, there were programming languages reaching end of support, or becoming too costly or risky to maintain.

The transition into the final phase thus came with an increased recognition of the aging of the mission infrastructure and an emergent appreciation for decline, loss, and finitude of infrastructural lifetimes. I have adopted the term "geriatric" to refer to this phase in the life of the infrastructure because it encapsulates both this recognition of aging and decline, and the ethic of care that became increasingly dominant during my fieldwork. Engineers increasingly worked to manage and negotiate the effects of aging and decay or what engineers called "lifetime issues" - bugs or anomalies that arise simply because over such a long life the unlikely becomes likely or the idiosyncrasies of long-term use crop up unexpected frictions. Indeed, engineers spoke of the spacecraft quite often in terms you might use for an elderly loved one, and at times felt a sympathetic connection to its aging.

The recognition of the infrastructure's agedness did not come all at once, but through a series of such moments of decay, which brought about a sense that the "machine is not what it once was" and so demands greater care. Yet at the same time, the transition was quite marked. Before the start of the final phase of the mission there had been a hopeful account of how the excitement of flying "closer than ever before" to the planet might lead renewal and rejuvenation - receiving a ramp up in funding, attracting a fresh crop of young engineers, and even adopting newer more agile development processes to match the iterations of their science planning to the faster rhythm of its orbits. After the start of the final phase of the mission, however, this excitement had faded and was replaced gradually by a sense of the need to slow down in the final phase in order to make it successfully to the end in 2017. The funding for a ramp up did not look likely, and besides, the engineers pointed out, the spacecraft was just not built for agility.

A geriatric infrastructure is thus one that is not merely old but is in the process of becoming old, with an emergent recognition of aging as a process and an increased appreciation for decline, loss, and finitude of lifetimes of the different parts of the overall system as well as the ways in which these multiple lifetimes are entangled. Decay and aging are a form of infrastructural change that can disrupt practice, but as I found, how such changes are managed and negotiated also involves a shift towards what I call convivial decay. This is a recognition of the negotiated nature of decay and the need to work actively with decline rather than against it, by attending to the mutual livability of its constituent parts and attendant practices of care.

The following sections offer vignettes from observations in the field that reveal the emergent recognition of agedness and acceptance of loss and decay. They also illustrate the negotiated nature of decay. While not exhaustive of the types of decay I observed in the field, they aim to provide representative illustrations of the work that engineers perform to bring their own work into alignment with a decaying infrastructure. These vignettes illustrate how engineers working to maintain and sustain this infrastructure came into an appreciation of its decline, coming to know/perform the agedness of the infrastructure, and actively work towards its end. I selected these particular vignettes with an aim to show the work of managing the aging and decay of both hardware and software, each of which had its own (in)visibilities within the mission organization.

From Kills to Quiet Time

During December 2010 while I was out of the field for the holidays, the Mission experienced an "anomalous event". The spacecraft received an incoming file and responded by going into "safing". The spacecraft could not interpret the file it received and so, instead of conducting science, it pointed towards earth, sent out a distress signal, and awaited further instruction. This of course led to some amount of concern on the ground.

The safing had come after sending a non-routine file to the craft that included a software update to the spacecraft's "flight software" – something that has only been done with great caution a few times over the life of the mission. This type of error would be difficult to repair since it could impact the ability of the engineers to communicate effectively with the craft to troubleshoot and locate the cause of error.

Fortunately it turned out that the error in the file was the result of a "bit flip" caused by the rare occurrence when the radio waves transmitting a file to the spacecraft are bombarded by a cosmic ray, turning a zero into a one and making the file unreadable to the spacecraft. This was fortunate because it meant that no human error had caused the safing, and that the problem could be repaired by simply resending the file. Of course, this still resulted in the loss of

science, not only the sequence of science that should have been received alongside the software update, but also the time lost as the engineering team worked to analyze and resolve what had occurred.

Gwen, the lead engineer for the spacecraft office told me afterwards that the anomaly had been a kind of wake up call, a moment of recognizing the limitations of the spacecraft as well as her own. Gwen had a kind of sympathetic relation to the craft. Around the time that the spacecraft wheels had started to deteriorate she had some health issues resulting in surgery on her leg, affecting her mobility. With the safing, she had experienced the threat of an error introduced into the spacecraft hardware as a moment of realizing that her memory might not be what it once was and that she might need to rely more on the next generation of engineers rather than her own "gut instincts."

The safing was felt by Gwen as a reminder of her own aging and the aging of the craft together, a wake up call to put some procedures into writing for the sake of the final years of the mission. She recognized in this moment, a need to write down more, to capture her expertise in order to transition some leadership to a younger engineer protégé. But it was also framed, in retrospect, as a kind of gift of time — more time to figure out what is going on with the spacecraft. I stopped by to talk to one of Gwen's engineers, Jared, who told me about the event. He seemed actively excited about the safing and how it had opened up a chance for "spacecraft activity" and "to learn new things" including testing out which of the reaction wheels on the craft was in worse shape.

This kind of time dedicated purely to engineering purposes is very rare. It coincided with some tension that had emerged around the degradation of the reaction wheels. The additional stress on the engineering team to implement science that did not cause harm to the wheels had lead to the loss of science. As I will discuss in a later section, this had caused a frustration among the science teams that science was being "deleted" or "killed" by the software used to protect the wheels or by the engineering team themselves.

Gwen explained this attitude of science being seen as "killed" as arising from an attitude that does not attend to the craft as a limited resource. "The science planning folks only think about how much science to fit in, doing more science." Sometimes there are segments of the spacecraft's orbit which are of less interest to scientists and Gwen's team will have a "quiet segment" when the spacecraft is essentially just flying, relying on its own automated software. Gwen sees value in *quiescence*, "but," she says, "the scientists don't understand that, they don't understand the idea of less science."

Now the safing event had caused a lot less science. When I stopped by the Science Planning area to catch up with Cassie, she also brought up the safing and I expected her to be frustrated, but she seemed to concur with what I heard from those on the spacecraft engineering team. She said:

"The safing was interesting because they learned about some communication mechanisms they can have in place to make things go better now that the organization is somewhat different. Gwen pushed really hard to have a longer time after the safing to recover, [in order] to manage people's expectations that they will not be always recovering from safing as quickly as they used to. Not that safing happens that often, like it does on [some other mission] where they happen all the time, but we've had a record of very quick recovery from safing and as we go into [the final mission phase] we may not recover as quickly."

This was a marked shift in the way that loss of science data had been spoken about before the safing and the transition into the final mission phase. The anomaly created a moment of pause, which was seen as a gift to the spacecraft engineering team to reassess their processes, but it also became a resource in managing expectations from the science team, and shifting them from an attitude of kills to one in which quiescence, pause, and slower responses may be necessary to give time to react. In turn Cassie was using this shift to help condition the scientists under her management to the idea that loss of science was not an opening for renegotiation of which science discipline would win in battles over future orbital segments.

This process of negotiation became a kind of tide change. Not a win for engineering per se, but a shift towards appreciation of loss as a necessity in this final phase of the mission and a refiguring of the relationship between loss and benefit. The anomaly was a moment of "pause" not only for Gwen but for the entire organization to reorient a kind of "infrastructural inversion"[4] when the needs of the spacecraft as an aging system became clear.

Making them say "Ow"

In the case of the spacecraft, a change such as hardware deterioration or software malfunction is powerfully visible within the organization. In the transition into the final phase of the mission there were other similar events such as an on-board instrument "shorting out". In each of these events, a very conservative response was taken that prioritized safety of the craft over the needs of science. This was often achieved by positioning the spacecraft as aged and in need of care.

However, when it comes to the mission ground system infrastructure, it proved much harder to assess, know, or make visible the agedness of the various software tools. If software decays it does so in ways that are largely invisible to the organization. One of the most glaring reminders of this was the fact that I heard many conflicting stories about what the transition to the final mission phase would mean for software systems at the mission.

Both before and after the start of the final phase I heard that

this phase would mean less science, due to cuts in staff and resources. However, there were much less consistent remarks about whether the transition would mean more or less software. I heard both stories from people all over the organization. Some said that the final phase would mean fewer personnel, a more fragile craft, and increased reliance on software to pick up the pieces and automate the work done by people leaving the mission. Others said that there would be less software in the final years, since as you lose people, you lose knowledge about how software works, and when a piece of software breaks and the person who wrote it is no longer around, you will have to go back to doing it manually.

Rather than a marked transition into acceptance of data loss and quiet time for the spacecraft, Sarah, the lead engineer in charge of the ground system had a difficult time relaying to management an appreciation of how software systems obsolesce and age.

"They don't really have much experience of what happens after launch… Once the system is perceived as mature, which in many people's minds is at launch, the perception is that it shouldn't need much attention. Which we know is not true, but [it] remains a widely held belief."

In an effort to demonstrate and communicate the needs for software maintenance, Sarah had been cataloging the extent of undocumented software on the mission and asking the various engineering and science planning teams to do similar "house-keeping" to generate lists of their most important scripts and utilities.[2]

We sat and talked in her office about the state of the ground system and the role of software in the mission operations, while she worked on preparing an inventory of all the "Category D" software used on the mission. Category D is not even a real category, but a catchall term for all the software tools that usually go undocumented.[3] While Category D is, by definition, meant to be software that engineers can "live without," in reality these are scripts and small programs that have accrued in the hundreds over the many years of the mission.

"Have you ever heard the term glueware?" Sarah asked me, "Well you start with this system design from 15 years ago, and as you go along you have to tweak it, but it's too hard to go into the compiled stuff, so you start adding on the inside and the outside." While Category D is not officially delivered software, in the many years of the mission these scripts have proliferated and become part of the fabric and "glue" that keeps the infrastructure going.

I brought up with Sarah this inconsistency about whether there would be more or less software in the final mission phase. "Which is it?" I asked. She responded with exasperation, as paraphrased in my field notes quoted here:

At this Sarah sighed and went somewhat silent and reticent and was clearly expressing a LOT with that silence. There was some emotion in it, perhaps some sense of awe for the problem itself and giving of silence for the people whose lives are impacted by this problem. It may also have been anger, letting some anger come up and settle down before speaking and composing a way to speak honestly and directly but also diplomatically. She said, well, she has been in that argument for a while now and it seems like the decision is that we are going to have more software than we can afford to maintain.

She went on to explain further that when they originally planned out the resources for the ground system… in terms of IT people, programmers, etc. to complete the work and maintain the software, [for the final phase of the mission] it was based on assumptions that have turned out to be false… but that the decision is just to proceed according to plan anyways.

She was especially silent at this point saying, well, what is going to happen I think is... that things will have to slow down. She said this slowly. Slow. Down. When something breaks, when you have a change request, it will take much longer than it does now and longer than you want it to.

Like Gwen, Sarah was working to prepare both her group and the Mission as a whole to the idea of loss, particularly in the few months remaining before she retired and there would no longer be anyone in her position. The transition to the final mission phase entailed a lot of software changes at once as efforts to reduce costs by migrating from versions of software that the Lab's centralized software team no longer supported or to reduce overhead costs associated with maintaining software.

One such change involved migrating from an old file sharing system to another. In preparation for a meeting where the various teams would report back to her on their "homework" of cataloging their own scripts and Category D software, she also moved a single file test from the old to the new file sharing system. At the meeting Sarah presented the results of her survey of Category D software. She then asked for a raise of hands from everyone whose workflow was affected by the one moved file. When everyone's hands went up there was a gasp in the room followed by uneasy laughter. There was an uncanny feeling that arose in the

[2] Software scripts are snippets of code that can be used to automate a small part of a workflow, e.g. watching a repository for particular incoming file updates and sending an email. Utilities were the colloquial name for small software programs that are perceived only to affect quality or ease of work, such as a tool that color codes spacecraft data to ease routine monitoring of subsystem status.

[3] Ordinarily the organization only keeps an inventory of the software tools that are considered "mission critical" according to standards provided by NASA so that changes to these systems receive proper oversight. Category A software, e.g., are those tools where an erroneous change to the code could result in mission failure.

realization that the software that each team uses is not really all that distinct but is bound together, often through "glueware" that links them all together. "I can't believe it touched all of us" one engineer said. Sarah called this technique, making them "say ow" – giving them a single pinch so that she could save them from more dramatic pains later on. Without the aide of phenomenal events like an anomaly or degrading wheel, Sarah had to manufacture something to trigger recognition of the ground system's "maturity".

Degrading Science vs. Degrading Hardware

As mentioned earlier, the safing event on the spacecraft had given the engineers time to test out one of the reaction wheels that had started to shown signs of decay.[4] These wheels help to orient the spacecraft in space as it orbits Saturn. The movements of the spacecraft must be finely "choreographed" since the fields of view of its scientific instruments are quite small and the spacecraft itself is hurdling by these targets at many kilometers per second. Science planners use a piece of software called the Pointing Design Tool (PDT) which spits out a command file for pointing the spacecraft in space, but are largely unaware of how these designs might affect the spacecraft as a whole.

The engineers in charge of the articulation control system are continuously monitoring the wheels, comparing incoming telemetry data to their expected behavior. These engineers must carefully examine and tweak these command files to protect the safety of the craft – making sure that e.g. instruments do not overheat from pointing towards the sun, or that the spacecraft is not sent conflicting commands to be in two places at once. They can sometimes protect the intent of the pointing by maintaining the orientation of the spacecraft along one axis while offsetting it along another. They can also manage the wheel spin rates (keeping them from reaching upper bounds) by inserting "biases" where excess wheel speed is dissipated by using thrusters (and precious fuel) to stabilize the craft.

These engineers place biases by using a piece of software called the Reaction Wheel Bias Optimizer Tool (RWBOT). This software models the spin rates needed to achieve the required pointing and optimizes for fuel use, suggesting some solutions for where to place biases into the sequence. The engineers try to place these between scientific observations but occasionally have to cut some science to make sure the wheels are safely commanded.

But now these engineers noticed that the wheels were experiencing some unexpected drag suggesting that something was wrong.[5] To research the cause of this drag, the Mission organization called in wheel rotation dynamic specialists, and consulted the wheel manufacturers. The wheels were specified by their manufacturer to have a guaranteed number of turns in their lifetime as long as they were kept within upper bounds. However, the Mission organization soon discovered that the wheels likely cause of decay arose due to dwelling in very low speeds close to zero and shifting subtly back and forth around zero. They theorized that this might cause the lubricant around the wheels' ball bearings to clump such that the wheels might eventually seize up.

This research was then integrated into RWBOT, which was given new parameters to avoid lower bounds and to balance fuel use with demands on the wheels, which were now also considered a "consumable" like fuel. The results of this change were dramatic. The engineers in the spacecraft office were no longer able to easily arrive at solutions that would protect the wheels. The software itself was taking much longer runtime to arrive at proposed solutions. Engineers would set it to run overnight while they were off work and return in the morning to analyze the results but there was just insufficient time in the schedule to run and re-run the software if the proposed solutions negatively impacted science. Meetings where the spacecraft engineers presented RWBOT's solutions to the science team became heated and tense. Spacecraft engineers were accused of "killing science" and science planners questioned the prioritizing of the wheels asking whether their degradation should be staved off by "degrading science".

This was not the first time that RWBOT had been a source of tension between the two teams and old debates were trudged up about the ways that RWBOT mediated communication between the two teams or how the spacecraft engineers handled the translation of science pointing to protect the wheels[6]. In the past, the tensions had been resolved through the creation of new rules. If very long observations tracking a slowly moving object like a moon had been bad for the wheels, a new rule was instituted that restricted the length of observations.

But with the decay and its resulting change to the RWBOT software, the engineers struggled to derive any predictable problems from their experience using software in its new

[4] The spacecraft has three "reaction wheels" which are spun in order to rotate the spacecraft in space along three (x, y, and z) axes. Because the spacecraft is floating in the vacuum of space, it is a change in the rate of spin of the wheels that results in a rotation of the spacecraft.

[5] There are legitimate reasons why the wheels might experience drag, such as when flying close enough to a moon to experience drag from its atmosphere. The spacecraft is given "torque authority" to autonomously override incoming commands in order to achieve the commanded wheel spin rate and overcome such drag. But when the drag on wheel 3 was spiking unusually high, the spacecraft office realized that they needed to re-assess the health of the wheels.

[6] For more discussion of how software tools mediate collaborative work in this case, please see [18].

version. For one thing, they had not had time to develop a sensibility for the kinds of solutions that RWBOT was now choosing. Eventually they created some "guidelines" being careful not to create rules that might suggest that the behavior of RWBOT was somehow predictable. They titled these guidelines as how to be "RWBOT-friendly."

Eventually an RWBOT meeting went smoothly with no deleted science. There was no clear singular cause for this adjustment but engineers spoke to me about their theories. One was that the difficulties with RWBOT coincided with a change in the inclination of their orbit, which changed the geometries of scientific observations and put more strain on the system. Another was that there was a shift towards a more "friendly" attitude among the scientists towards the new guidelines. In the past, rules instituted for protecting the safety of the craft had defined keep-out zones which had been hard coded into the PDT software such that scientists tended to work around them, doing their best to "game the system" (e.g. if an observation must be shorter than a particular duration, creating a long observation by dividing it up into sequential shorter ones).

While this worked for most kinds of rules developed by engineers so far (e.g. in the form of keep-out zones where instruments cannot point), the guidelines to protect the wheels were there mostly to help the engineers make their work rhythms sustainable. The new RWBOT-friendly guidelines were written to enforce a more holistic perspective, which required scientists to view RWBOT less as an unpredictable "troll under the bridge" that kills science, and more as one tool used by their engineering colleagues.

As a moment of celebration at the end of the first smooth RWBOT process, Gwen wheeled in a giant birthday cake to deliver as a gift to the science planning team, as a form of recompense for lost science. She jokingly explained the gift saying "now you can't say we never give you anything" responding to the sense that engineering only takes away science rather than supporting it. The cake decoration spelled out "5000th day since launch", marking a birthday of sorts of the spacecraft. This highlighted the age of the spacecraft, but also made visible the everyday temporal rhythms of engineers in the spacecraft office where "days since launch" are counted. While the science planning team was on their 50th science sequence, the spacecraft team who works on a daily basis to command and operate the craft count out time differently.

Letting Software Break

While these changes to expectations of the spacecraft hardware arise due to material decay, there is no real catalyst for making people see the pain that might come from relying upon unmaintainable software. Due to the decrease in staffing, the software team was by far the most reduced in size, and in fact had already been cut significantly before the middle phase of the mission. Sarah was concerned in light of her retirement that it would be an issue related to software rather than hardware that might bring the mission abruptly to a premature end.

In preparation for the final phase, the team was letting go of permission structures that provide a layer of protection from errant changes to the software. The software team was removing its separation of designated operations and development servers – switching to a reliance on the developers themselves to keep their development work from interfering with software in active use. They also were making reductions to their "configuration management" such that everyone would have access to all servers and would have to rely upon trust and communication to not disrupt each other's work.

Learning to live without these divides and partitions was a difficult transition. Downgrading and letting go of software systems feels counter-intuitive since software that works and runs well becomes invisible. At a meeting that Sarah held to go over these changes, she went through the new set up, going through an excel list of machines and who was responsible for each. At first the meeting proceeded in a typical fashion, with Sarah and other managers overseeing the decision-making of how these servers would be configured. Suddenly, the discussion shifted as individuals committing to particular set ups realized that this was going to be the last managerial discussion of its kind. They began to recognize that this configuration was a different sort of task than they had done before. With the new system, people might be able to get into a server space that they could not before. This would not result in changes in permission structure but would have to be handled in an ad hoc manner.

Sarah actively shifted the conversation, moving from possible scenarios posed in terms of "suppose that something breaks" to an assumption that breakages will happen, by asking "how would it break?" When her colleagues scrambled to figure out a way to set up the permissions so that they would not break, Sarah reiterated that "it *should* break" and repeated this throughout the conversation until others began to concur, saying we should "go ahead and set it up knowing it is going to break."

The discussion started to shift and people began to joke about how they should "speak now or forever hold their peace." When it was decided that it would simply be the user's responsibility to know the purpose of each machine and to not make changes without talking to the person responsible for the server (literally with the server at their knees), one developer in the room captured the unsettled feeling in the room calling out "It's the wild west!" and someone else chimed in "Enter at your own risk" to which everyone starts laughing, breaking the tension.

This new Wild West mode in which you enter at your own risk is one where they give up tight bureaucratic control in favor of "a common machine", trusting in the collective

behavior of the group to work itself out and trusting breaks to guide misdirected attempts to the proper machine. It denoted a shift towards a mode of thinking about breaks as productive. Software should break, and stay broken, as a way to ease users off of unmaintainable systems or protocols. This reframing is similar to the role played by the spacecraft anomaly, in which people are conditioned to accept loss, adjusting to more and more breaks in systems associated with the reduced resources of the final phase and coping with these changes.

But more than merely acclimating to loss as a new state of affairs, this effort required an active working towards loss, working to let go of system-entanglements, working towards breakages as a productive process of letting go. In letting go of particular partitions, separations both material and metaphorical, the relationship between letting go and moving forward is exposed. Breaking things is needed in order to extend the possibility of keeping them going. In a way breakage is for software what quiescence is for hardware — a respite.

DISCUSSION

The case of a geriatric infrastructure offers a novel empirical example of infrastructure repair where there is a known end-of-life towards which engineers are working. In such work, there is ongoing work to extend the life of the infrastructure to ensure it reaches 2017, and of course the work of engineers is filled with novelty and inventiveness even at this late stage in the infrastructure's life. In this case we can see familiar forms of work including repair-as-sustaining and repair-as-transformation [25] as well as repair-as-appropriation [16] or repair-as-sustaining [11].

However, the case also identifies a related empirical form of work that is repair-into-decay. Repair work can at times work not to renew but to let go. Rather than fending off death [8], repair can embrace endings, finitude, and loss, working with and into decline rather than against it. The case contributes empirical understanding of the careful and cautious work that is involved in disconnection and disassembly [8]. It also reveals the collaborative work through which such decline is performed, managed, and negotiated.

In the vignettes shared above, we see that repair-into-decay involves an increased appreciation for loss, decline, pause, and finitude. This appreciation appears to emerge on the one hand in response to ruptures in the form of anomalous events that trigger new understandings of the infrastructure as aged and in decline and requiring of adjusted expectations. Yet on the other hand, we can see how the relative agedness of various parts of the infrastructure is actively performed as part of the negotiation of what and how this decline should unfold.

In the following sections I discuss how we might make sense of this seeming contradiction. Agedness is both recognized and performed and even the ruptures which trigger an awareness of agedness reveal a doubleness in which infrastructure is old and new at once. Repair-into-decay invokes a negotiation of what counts as old, what should survive, what should persist, and what should be let go. This negotiation is one that attends to the mutual livability of work practices in a form of alignment work I call convivial decay.

Recognizing and Performing Agedness

In the vignettes I have shared, we can see an ongoing negotiation of what will degrade and how as the space science infrastructure is increasingly recognized as aging and in need of care. In some cases particular changes due to decay, such as the wheel degradation, are events that disrupt current practice and surface tensions across the work practices of different teams, which need to be brought back into a sustainable alignment. While disruptions to current practice are not necessarily new to this stage of the mission, what is notable is that these events trigger, or are taken up as resources for an emergent recognition of the "stage of life" or "maturity" of the infrastructure and appreciation for loss, decline, and finitude.

There is a marked shift from a culture in which science and engineering are at odds with each other, with a language of killing and deleting of science in favor of hardware, to one in which loss, pause, and quiescence are perceived as beneficial and necessary. In some cases moments of changes in the performance of the spacecraft hardware make the aging and decay of the infrastructure imminently apparent. In other cases, it takes the active insertion of a disruption such as the movement of a file or the removal of a permissions structure, to trigger this recognition that the maintenance of the infrastructure requires some adjustment.

The recognition of the agedness of the infrastructure is not a process that isolates aging to a particular device. Infrastructural decay is a process that emerges through the ways that different parts of the system, aging at different rates are entangled with each other. What decays or ages are the *relations* across multiple parts of the infrastructure and among people, the organization, and its technologies.

For example, we saw how the sympathies between Gwen's own aging and the aging of the craft became a resource for her to ensure the ongoing life of the infrastructure by facilitating intergenerational hand-off and not assuming a fast and agile recovery from such anomalous events. As another example, we can see how the degradation of the wheel cannot be isolated to hardware or software or practice alone. The wheel degradation is in fact unknowable without the mediation of software used to analyze its behavior. It is not the degradation of the wheel itself that initiates the re-negotiation of work between engineering and science planning teams. Rather, the RWBOT software reveals how the management of wheel degradation and the interface between the two teams are simultaneously and co-constitutively negotiated.

Engineers who have worked for much of their careers on this particular infrastructure have developed sympathy akin to what Vertesi observed with the Mars Rover engineers in [32]. In this case, it is a sympathy with the aging of the system, experienced as an alignment of coming to know one's own aging, as well as the aging of the organization, through the aging of the infrastructure. The desire to rejuvenate what is perceived to be a "dinosaur" infrastructure past its prime through the adoption of organizational methods that are more contemporary with other missions, exemplifies how agedness is not simply known through physical decay but also through the ecology of different systems aging in relation to each other.

In this way aging is clearly a sociotechnical process where aging cannot be isolated to hardware or software, material decay or changes in practice. Furthermore, agedness not only arises across collectivities of people and machines, but is also performed through negotiations of what counts as old or the relative agedness of different parts of the system. In one moment what is figured as old in order to demand care, is figured as new in order to demand prestige.

Lifetime Issues

The recognition of agedness that emerged during my fieldwork was often catalyzed by events that disrupted practice as usual. As one engineer put it, this is the stage of life of an infrastructure where "lifetime issues" start to crop up. This can be in the form of a bug that emerges in software that has been used and maintained for decades, simply because of the idiosyncrasies of a particular orbit or of a new inexperienced engineer arriving to the mission and being "thrown into the deep-end of Unix" which is no longer taught in schools.

An event such as an anomaly on the spacecraft, a bug in code, provokes reflection on changes in the infrastructure that have arisen due to aging and decay. Lifetime issues are a type of sociotechnical change that occurs simultaneously as a rupture and a drift. They arise in a particular event but in the form of a post hoc reflection on something that is already latent in the system. Lifetime issues hold a doubleness of potentiality and things running their course.

This is seen quite evocatively in the case of the wheels, which clearly have been decaying gradually from the moment they entered space. The degrading wheels here are indeed a form of slippage [26] between understandings of the wheel's performance from manufacturing specifications and their actual performance, but the slippage itself is not something which can be located at a particular moment in time. The lifetime of a wheel is specified by manufacturers through simulations during testing – e.g. spinning the wheel fast until it over heats, spinning it for a long time until it shows break down. But the actual lifetime of a wheel-in-use imparts a different notion of that duration.

Even the anomaly, which is clearly experienced by the organization as a rupture, is, in retrospect, also understood through the lens of duration or lifetimes. The anomaly created a moment of pause in which the agedness and decay of the infrastructure were negotiated in terms of infrastructural, biographical, and institutional rhythms [14] including inter-generational hand-off. But the anomaly was also narrated to me as a "lifetime issue" in the sense that cosmic rays are of such a rare occurrence in the vacuum of space that the chance of bombardment of the Mission's radio wave communication to the spacecraft goes up the longer it is in space. "In fact," Gwen exclaimed, the infrastructure, as a whole, had become "a quite fine-tuned instrument for detecting the prevalence of cosmic rays in the galaxy."

These changes can be seen as a form of infrastructural torque [4,33]. Bowker and Star describe torque as a "twisting of time lines that pull at each other" as "trajectories" of institutions, categories, biographies, etc. "pull or torque each other over time [as] they move in different directions or different rates" [3]. As they point out, this twisting arises over the duration of lives. While there is perhaps a singular moment in which the engineering team at the Mission realized that the wheels might be decaying, this decay arises though only gradually. The twisting is also "multiple" in a similar way as referred to by [30] because infrastructures are comprised of multiple temporalities and trajectories of careers, hardware and software systems, managerial methodologies, etc.

"Lifetime issues" are precisely those issues that are happening in the background but only surface over the lifetime of a system. There is a coincidence within the case described here between the perception of increased occurrence of such lifetime issues, and the recognition of the need to slow down and learn how to appreciate agedness, loss, and decay so that these can be responded to appropriately. It is not clear which comes first – the need to slow down precipitating a recognition of lifetime issues or vice versa. I strongly suspect that there is no particular time in the life of the Mission or of any infrastructure project where there are significantly more or less ruptures that break the flow of routine work. Yet, these particular ruptures in the form of "lifetime issues" allow us to see how the performance of agedness participates in the process of negotiating how the organization responds to decay.

Declaring the infrastructure an instrument for detecting cosmic rays, for example, places it on the frontier of engineering and scientific knowledge, even in a moment of decay and breakdown. This performance of presenting what is old as new arose many times throughout my fieldwork. In a poignant moment, sitting with Gwen in her office, she related how the scientists want them to take science that was designed years ago "off the shelf and dust it off and then fly it on a new machine". In this, Gwen positions the aging spacecraft as new, and the science that exists in a design document, as old, inverting the traditional maintenance-design relationship. As the vignettes shared

above reveal, Gwen is successful in her efforts to gain more resources and time for her engineers who maintain the spacecraft in part because she is able to claim that the machine is new, it is not what it once was.

Convivial Decay as Alignment Work

The negotiation of decay is a form of alignment work to deal with the slippages [26] that arise between what a system once was and what it has become. However, what this case reveals is that decay occasions different forms of alignment work than have been previously discussed in the CSCW literature [e.g.2,14,28].

In the vignettes above, we can see forms of negotiation which are familiar to CSCW research, working through negotiations across scales [23], rhythms [14,22,28], and through the use of boundary-negotiating objects [16] such as the RWBOT software itself. Yet the negotiation also takes place through positioning the relative agedness of various parts of the infrastructure.

Tensions indeed arise across scales and rhythms - indicative of the different lifeworlds that are brought into conflict. For example, in the fraught RWBOT debates over what should degrade, it is the ways that RWBOT represents temporal scales and rhythms of work that has historically led to so much miscommunication between teams who have different ways of working. However, in the resolution of the present RWBOT issues, these differences were not the most salient tensions that needed to be resolved so much as the mutual livability of these lifeworlds side by side.[7] The different temporalities did not need to be brought into sync with one another but rather simply needed to be ground into a companionable relationship.

This may help to elucidate why a satisfactory explanation for the resolution of the RWBOT case remained elusive or why a cake can matter as a form of sociotechnical repair. They reveal that what is negotiated in the face of decay is a mutual livability of systems and practices that are running out different lifetimes, aging at different rates, and yet mutually entangled and interdependent.

Decay has a temporal quality that disrupts the assumed progressive temporality of technological change. It asks: were prior alignments as livable as we thought? Livability thus deals in rhythms but also in durations. It requires a consideration of which rhythms of work are sustainable or livable, and an awareness that the lifetimes of systems are themselves finite. The negotiation of decay is a negotiation of multiple lifetimes that are entwined – how to carefully cut these away from each other, or allow them to be companionable. It is this process of alignment that I call convivial decay.

Convivial decay is a form of alignment work that deals with the lifetimes and mutual livability of the relationships that inhabit infrastructure. It is a working with rather than against loss and finitude and an active letting go to enable livable sociotechnical worlds that are companionable with each other. It is an active pursuit to let go with care and maintain livability of different forms of work and different parts of the system.

Convivial decay is not something that just happens whenever a system ages. Clearly it takes work. In some ways the evocative decay and breakdown of hardware (the wheel's drag, the safing event) make it easier for the aging of hardware to demand attention. Software's aging is, however, less obviously visible and more counter-intuitive to management. The idea that a bug can surface through the idiosyncrasies of practice after decades of use is taken as quite shocking. It shows the same doubleness of how the software can be both old and new at the same time just as a decaying wheel can tell us something new about long-standing practices. Software developers seem to know this. It is inherent in the idea that the entire mission is a run time test of the software. Each iteration of science is unique and so each iteration holds the potential to break some long-held assumption embedded in the software. Discrepancies between the manufacturing specifications of the wheel and its actual operations are in many ways equivalent to the discrepancies between the design specifications of software and the practices of its use. Yet these software discrepancies are harder to know in a culture that treats software's histories as irrelevant or its life as stagnant after launch.

While the vignettes related to hardware above are in some ways more evocative, it is the relationship between software and conviviality that I wish to pursue in future work. Software is central to convivial decay in that it both contains its own processes of aging and mediates breakdowns of hardware and how these are made accountable and available to repair [29]. In the end the conflict over the spacecraft's wheel decay became a conflict over the way that a software tool, RWBOT, played a decisive role in what science lived or died. Software too has its forms of living and dying and its ways of mediating what forms of work are made sustainable and livable and which others are not.

In a way, it is software where conviviality gets complicated. This requires a lot more research - to understand how software ages, and how its multiple entangled lifetimes and their aging can be negotiated with conviviality. Particularly, when we consider that no infrastructural systems emerge whole-cloth without reliance on older legacy code, we can see how conviviality might be worth understanding, even in early stages of the life of systems, to help bring the caution and care that attending to the mutually entangled lifetimes of all systems warrant.

[7] This notion of livable worlds is drawn from [9].

CONCLUSION

In conclusion, what this case of a geriatric infrastructures shows is that infrastructural decay is known and negotiated through an emerging appreciation for the multiple and entangled lifetimes of infrastructure. Infrastructural decay is thus not a passive backdrop, or a "natural" process of time's passing. The temporality of decay is not a singular trajectory that unfolds along an evolutionary timeline, but rather is heterotemporal and relational. This is in part due to the relativistic time of lifetimes and aging as noted by Traweek in [31]. We only know what is old by virtue of what is young. While it may only be a problem for coordination that one part of a technology moves at a faster rate than another, this difference will produce a kind of torque over time, one each of these parts ages. It is only once a particular rhythm or piece of technology is lived with for a long time that we can recognize what has become old. We live in the aftermath of decisions about standards, rules, etc. that then apply torque as they age.

This implies that in order to understand the lifetimes of infrastructure it may not be enough to study them longitudinally. Such an approach implies a singular temporality of the biography of an infrastructure as it is born, lives, and dies. But what this case reveals is the heterotemporality of infrastructural lifetimes, something which becomes very apparent in the cropping up of lifetime issues and decay, but which is no less present in moments of design "upstream". Change which can only be understood and appreciated through the lens of duration, are not the exclusive purview of old and aging infrastructure – though this seems to invoke particular insights for those working with them that can be ignored in earlier phases. This is a methodological gift because durations are always available to us at any moment in the life of an infrastructure since no infrastructure is born completely anew without embedded legacies and histories.

Finally, what I wish to draw attention to here is that the nature of negotiation across the tensions of lifetimes is such that what is negotiated is not only alignment for the sake of coordination but for the sake of conviviality. What does it mean for the material, infrastructural, biographical, and institutional rhythms not only be aligned but to be livable and sustainable for the foreseeable future? These negotiations do not just take place in time but across time – for example negotiating across generational gaps and paradigmatic gaps. This too is a form of invisible labor, of working to time various forms of labor to each other, such that they might break gracefully together.

ACKNOWLEDGMENTS

Many thanks to the anonymous reviewers whose responses improved this work greatly. This work also benefited from insightful comments in early stages of analysis and writing from Lilly Irani, Steve Jackson, Max Liboiron and Naja Holten Møller. This work could not have been completed without the open dialogue and mentorship into which I was welcomed by my interlocutors at the Mission and by the collaborative engagements of the Spaceteams Research Group led by Janet Vertesi and including Paul Dourish, Melissa Mazmanian, Matthew Bietz, and David Reinecke. This work was supported by NSF Socio-Computational Systems Grant #0968616.

REFERENCES

1. Eric P.S. Baumer, Phil Adams, Vera D. Khovanskaya, et al. 2013. Limiting, leaving, and (re)lapsing. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*, ACM Press, 3257–3266.

2. Matthew J. Bietz, Toni Ferro, and Charlotte P. Lee. 2012. Sustaining the development of cyberinfrastructure: an organization adapting to change. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, ACM, 901–910.

3. Eli Blevis. 2007. Sustainable interaction design: invention & disposal, renewal & reuse. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, 503.

4. Geoffrey C. Bowker and Susan Leigh Star. 2000. *Sorting Things Out: Classification and Its Consequences*. MIT Press.

5. Graham Button and Wes Sharrock. 1994. Occasioned practices in the work of software engineers. *Requirements engineering*, 217.

6. Paul Dourish. 2003. The Appropriation of Interactive Technologies: Some Lessons from Placeless Documents. *Computer Supported Cooperative Work (CSCW)* 12, 4, 465–490.

7. Harold Garfinkel and Harvey Sacks. 1970. *On formal structures of practical actions*. Appleton-Century-Crofts, Educational Division.

8. Stephen Graham and Nigel Thrift. 2007. Out of Order: Understanding Repair and Maintenance. *Theory, Culture & Society* 24, 3, 1–25.

9. Donna J. Haraway. 2003. *The companion species manifesto: Dogs, people, and significant otherness*. Prickly Paradigm Press, Chicago.

10. Sampsa Hyysalo and Janne Lehenkari. 2002. Contextualizing Power in a Collaborative Design Project. *Participatory Design Conference*.

11. Steven J Jackson. 2014. Rethinking repair: breakdown, maintenance and repair in media and technology studies today. In *Media technologies: Essays on communication, materiality, and society*. MIT Press.

12. Steven J. Jackson and Lee Kang. 2014. Breakdown, obsolescence and reuse: HCI and the art of repair. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 449–458.

13. Steven J. Jackson, Alex Pompe, and Gabriel Krieshok. 2012. Repair Worlds: Maintenance, Repair, and ICT for Development in Rural Namibia. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, ACM, 107–116.

14. Steven J. Jackson, David Ribes, Ayse Buyuktur, and Geoffrey C. Bowker. 2011. Collaborative rhythm: temporal dissonance and alignment in collaborative scientific work. *Proceedings of the ACM 2011 conference on Computer supported cooperative work'*, ACM Press, 245–254.

15. Helena Karasti and Karen S. Baker. 2004. Infrastructuring for the long-term: ecological information management. *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, 1–10.

16. Charlotte P. Lee. 2005. Between Chaos and Routine: Boundary Negotiating Artifacts in Collaboration. *ECSCW 2005*, 1–21.

17. Leah Maestri and Ron Wakkary. 2011. Understanding repair as a creative process of everyday design. *Proceedings of the 8th ACM conference on Creativity and cognition*, ACM, 81–90.

18. Melissa Mazmanian, Marisa Leavitt Cohn, and Paul Dourish. 2014. Dynamic reconfiguration in planetary exploration: a sociomaterial ethnography." *MIS Quarterly* 38, 3, 831–848.

19. William Odom, James Pierce, Erik Stolterman, and Eli Blevis. 2009. Understanding why we preserve some things and discard others in the context of interaction design. *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, 1053.

20. Julian E. Orr. 1998. Images of Work. *Science, Technology, & Human Values* 23, 4, 439–455.

21. Neil Pollock and Robin Williams. 2010. e-Infrastructures: How Do We Know and Understand Them? Strategic Ethnography and the Biography of Artefacts. *Computer Supported Cooperative Work (CSCW)* 19, 6: 521–556

22. Madhu Reddy and Paul Dourish. 2011. A Finger on the Pulse : Temporal Rhythms and Information Seeking in Medical Work. *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, ACM, 344–353.

23. David Ribes and Thomas A. Finholt. 2007. Tensions across the scales: planning infrastructure for the long-term. *Proceedings of the 2007 international ACM conference on Supporting group work*, ACM, 229–238.

24. Daniela K. Rosner, Steven J. Jackson, and Garnet Hertz. 2013. Reclaiming repair: maintenance and mending as methods for design. *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, ACM, 3311–3314.

25. Benjamin Sims and Christopher R. Henke. 2012. Repairing credibility: Repositioning nuclear weapons knowledge after the Cold War. *Social Studies of Science* 42, 3, 324–347.

26. Benjamin Sims. 2009. A Sociotechnical Framework for Understanding Infrastructure Breakdown and Repair. *Presented at the Annual Meeting of the Society of Social Studies of Science*. http://public.lanl.gov/bsims/

27. Susan Leigh Star and Anselm L. Strauss. 1999. Layers of Silence, Arenas of Voice: The Ecology of Visible and Invisible Work. *Computer-Supported Cooperative Work: The Journal of Collaborative Computing* 8, 1–2, 9-30.

28. Stephanie B Steinhardt and Steven J. Jackson. 2014. Reconciling Rhythms : Plans and Temporal Alignment in Collaborative Scientific Work. *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, ACM, 134–145.

29. Lucy A. Suchman. 1987. *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press.

30. Bill Tomlinson, Michael Silberman, and Don Patterson. 2012. Collapse informatics: Augmenting the sustainability & ICT4D discourse in HCI. *CHI '12 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 655–664.

31. Sharon Traweek. 1992. *Beamtimes and Lifetimes*. Harvard University Press.

32. Janet Vertesi. 2008. Seeing like a rover: embodied experience on the mars exploration rover mission. *CHI'08 Extended Abstracts on Human Factors in Computing Systems*, ACM, 2523–2532.

33. Janet Vertesi. 2014. Seamful Spaces: Heterogeneous Infrastructures in Interaction. *Science, Technology & Human Values*.