



Practical Considerations on Applications of the Popularity of Games: The Case of Location-Based Games and Disaster

Nicolas LaLone¹ , Phoebe O. Toups Dugas² ,
and Konstantinos Papangelis³ 

¹ University of Nebraska at Omaha, Omaha, NE 68182, USA
nlalone@unomaha.edu

² New Mexico State University, Las Cruces, NM 88003, USA
phoebe.toups.dugas@acm.org

³ Rochester Institute of Technology, Rochester, NY 14623, USA
kxpigm@rit.edu

Abstract. In the midst of a disaster event like a hurricane, all electrical, connected objects are typically rendered useless. A lack of connectivity, electricity, and potential mobility issues render devices (and sometimes users) unable to perform their basic functions. The potential for the sheer volume of these devices, of the apps installed on them, are as such that they are an unused canvas of design. We present extensible design, the activity of designing new uses for existing applications that may possess functionality that is useful outside of its intended function. We present a description of extensible design and provide a fictional example of what that approach may provide. In so doing, we help address existing gaps between emergency management and consumer-based communication behaviors during disaster. The “Decentralized Layer,” an extension of location-based games like *Pokémon Go*, *Pikmin Bloom*, and *Harry Potter: Wizard’s Unite*, is meant to provoke discussion about the potential use of apps and the app ecosystem past its current, limited expression. We conclude by offering next steps, road blocks, and additional considerations for extensible design that will need to be in order for it to be realized.

Keywords: Game studies · Design fiction · Extensible · Play · Location-based games

1 Introduction

Game studies has been in a perpetual defensive stance by simultaneously attempting to show that games were worthy of academic inquiry and that game studies was a field worthy of academia [1, 20]. Having accomplished the task of showing its worth, academia has seen the study of games spread to a variety of disciplines [20] and stabilize in the form of journals, grant funding, college departments, conferences, and special interest groups. The need to move past

proving its worth to using the knowledge game studies has generated is becoming more pressing. Small attempts at this have been successful through applications like gamification or serious games, which call back to Huizinga’s original debate with Carl Schmitt on the seriousness or lack of seriousness of play [40].

Others have used the tenets of play in an effort to foster games-based learning or actively use the making of games to teach programming through fantasy consoles like PICO-8 [66]; game engines like Twine [8]; programming languages like Processing [59]; and projects like Handmade Hero [47]. Yet, there remains an ever-increasing potential in games that extends past their internal components. The popularity of games and the sheer number of app downloads, physical consoles, and other crowd-based efforts remains a vast, untapped resource. Often in the 10s of millions, the use of these computational resources for contexts other than the various ways we play them could foster new, innovative ways to reconsider games, play, people, non-human objects, and culture in concert.

The contribution of the present research is to present a theoretical realm of design we call “extensible design”: the application of new capabilities to existing products. More than modding, hacking, or writing addons, “extensible design” is a way of seeing other products as spaces of play through which other types of development can occur. For example, popular location-based games (LBGs) like *Pokémon Go*, *Pikmin Bloom*, and *Harry Potter: Wizards Unite* afford the player location-based services, a routinized map interface, and certain types of communication tools [55, 56].

Each of the affordances of these tools can be used in other ways, for other reasons, and in ways that can benefit not only the player, but other players, places, and contexts [55]. For example, when disaster strikes, the affordances these games possess can be useful to emergency management, disaster response, and first responders whose job it is to understand where survivors might be and what their needs are – and they are available on an astounding scale. With extensible design, we envision ways that users can note their locations even if connectivity to the centralized network is severed. We display this vision through a design fiction [5, 15, 28, 36, 51, 61].

1.1 A Design Fiction for Extensible Design

Design fiction is the use of speculation to outline, display, criticize, and explore potential futures as a way to facilitate debate and discussion around current modes of design [5, 15, 28, 36, 51, 61]. Through this concept we outline the creation of an “extensible design” meant to harness the popularity of location-based games as a safety tool given devastation, destruction, and even connectivity loss.

We use “extensible” languages in computer science like lua or, at least, extensible grammars like Perl6, Lisp, Red, and Racket [70] to inform this definition. By extensible, we mean that, within these languages, it is possible to move beyond the limitations of a controlled environment. Programmers can instead create emergent types of structures or syntax that extend the operation of these products to places that the creators and maintainers of the languages did not expect. This metaphor is appropriate as we envision this concept as an extension of what already exists as understood through the task-artifact cycle [11].

Extensible design is additionally tool design using existing tools. It could also be referred to as appropriation [23] or a “design for hackability” [29], though we prefer to think of it as an extension of the task-artifact cycle [11]. The basis of this tool is the standard array of affordances that location-based games possess: friend lists, map-based play, location-based services, and the use of that location on an accurate map. And through this tool, our design fiction begins with a prompt that asks, “what would it take to make a mobile device or internet-enabled device useful in the circumstance of no power, no connectivity, and no ability to communicate?” This prompt is one central to current issues in the domain of crisis informatics [53].

Crisis informatics, or the study of information flow during disaster, has been attempting to harness the sheer number of mobile devices in the midst of crisis for over 15 years [53, 63, 65]. There are multiple bottlenecks to this research, including a lack of computer science presence in emergency management in any meaningful capacity. The most prevalent bottleneck is that, in the midst of crisis, data lines do not function, electricity does not function, and, without these things, all those mobile devices, tablets, and computers are functionally worthless. This does not have to be. Instead, we develop a design fiction around a fictional, but buildable and relatively inexpensive, Bluetooth transponder. This hardware can be used to create and extend a decentralized mesh network. Through that decentralized network, extensible applications can be created that work only on that type of network. These applications are the focus of this work as these apps can be created to work without electricity or connectivity in mind.

To outline this fictional product, we begin by discussing the act of design with specific attention to the task-artifact cycle [11]. The task-artifact cycle, in this case, is the push and pull of designers creating an artifact to perform a task and users engaging with that product in ways designers did not intend, thus forcing the designers to change the artifact and restarting the cycle. After discussing the task-artifact cycle, we begin to provide the foundation for our design fiction, “The Decentralized Layer” by focusing on its needs, its audience, and its intended use. So founded, we then move on to the components, its parameters, and its potential affordances given what it is extending.

“The Decentralized Layer” is a new use for LBGs manifested by a push and pull between a lack of connectivity and the affordances of LBGs. We offer that transponders can create a battery-powered mesh network that mobile devices can log onto as long as those mobile devices have additional capabilities to harness that local network. These transponders would be inserted into disaster-proof boxes around populated areas that will activate given a lack of electrical current. Upon detecting this potential signal, the extensible applications that exist within popular apps like LBGs, will activate. Using low-energy mode, users might share low resource messages that are geo-coded (e.g., they are trapped and need help, they are assisting others, they are on the move) in order to provide useful information to responders as they enter ground zero.

While this network is active, drones and other equipment that can reach populated regions faster than responders can pull on the meta-data of these

applications [45] and feed it to the search and rescue (SAR) teams that will be among the first to arrive to begin to treat others. In offering location and brief descriptions of where survivors are, SAR team members can have an understanding of where humans are and what their needs might be, focusing effort and reducing the need for wide search patterns [38]. In offering this fictional account of a product at the beginning stages of development, we provide a way to not only extend game studies to a practical space, but offer different ways to think about the data generated during a disaster.

2 Background: The Task-Artifact Cycle

This section outlines the design process as it relates to the task-artifact cycle in human-computer interaction (HCI). We begin by discussing a series of decisions made in the foundation of modern computer science. After this, we will discuss scenario-based design and various iterations of user-centered design. Finally, we will connect play to the act of design and move toward extending the task-artifact cycle or “extensible design.”

2.1 Whence Does the Task-Artifact Cycle Come?

In order to understand the task-artifact cycle, it is important to situate it in the history of computer science and the design of software. Much of the task-artifact cycle can be followed back to the creation of ALGOL60 [4]. The creation of this computer language was significant for 3 reasons. First, through the work of Peter Naur, Edsger Dijkstra, and Jaap A. Zonneveld, this was the first language that allowed for recursion [16, 21]. Second, this language was the first to be compiled and run as software [37]. Third, this was a language that moved the creation of software to be generalized whereas most languages and machines at that time were specialized [16]. The creation of ALGOL60 [4] and the events surrounding Peter Naur and Edsger Dijkstra [49] after its creation help us understand the origin of the task-artifact cycle [11].

Naur, who had played a part in the creation of some of the earliest computer programming languages, believed that individual perception of human-based issues that computers could solve were important. “Programming as Theory Building” was a treatise to this effect [48]. In this article, Naur outlines a concept of design wherein a group of programmers create a program to mediate human activity and stick with that program as it represents the end result of building a theory of human activity [48].

Dijkstra, on the other hand, believed that science, that objective reality was a requisite factor in the creation of software [22]. To Dijkstra, “short is beautiful” was a way to show that only elegant, easily replicable programs should be sought [18]. What this provided was not an understanding of human activity, but of a problem that could be solved mathematically [18].

What this essentially builds toward is a representation of the goals of programming that has expanded to the current day. Naur, on one side, sees a computer program as essentially a theory of human activity that must be continually

adjusted [49]. Dijkstra, on the other side, sees a computer program as the shortest answer to a question [17, 18]. This binary is not simply one between Naur and Dijkstra, but is often seen within the tensions between qualitative research and quantitative research, natural science and social science, or human factors and HCI.

The consequences of this clash and the subsequent victor can be seen in the various paradigms of HCI [24]. As HCI separated from human factors engineering, HCI inherited the search for simplicity in programming as well as its initial drive to represent humanity through the “elegance” of mathematics [10]. This was translated by early designers as it developed into user-experience engineering, user studies, and user-interface design [24]. Its codification and stabilization was the creation of the task-artifact cycle [11, 14].

2.2 The Task-Artifact Cycle

To wit, software is typically the briefest answer to creating an artifact for a task. For example, *Microsoft Word* revolves around the idea of writing. *Microsoft Excel* is a spreadsheet program that allows users to organize data in rows and columns. Whereas answers to questions like, “how can software replicate writing?” the answers themselves, once deployed as software, encounter the pluralism of humanity [17]. Tasks are rarely static and computer programs themselves cannot adjust to meet how a task changes unless re-deployed, patched, or replaced. As a result, programs like *Microsoft Excel* and *Microsoft Word* evolve, the task changes and in turn, the program changes.

Users will appropriate [23, 25] *Microsoft Word* as a photo editing program and so attention is paid to affordances that allow users to edit photos in some way. *Microsoft Excel's* task has extended to charts, graphics, and statistical measures. And so, Microsoft's designers have had to adjust *Microsoft Excel*, affording users more and more of an ability to use *Microsoft Excel* as one would use *SPSS*, *R*, or *SAS*. This is the task artifact cycle, a piece of software is created for a task and this locks a company or designer into a consistent pattern of design and re-design as the task changes due to users understanding a program [11]. We can see this process at work through patches, versions, and relaunches.

Yet, within the constant barrage of change, there is an inherent weakness in the task-artifact cycle. While tasks evolve, we must always have a set task that we can afford users to perform. This is a bounded space, a magic circle [32] that must always exist as its current version does. Each new version re-situates the circle, reboots the space for a new version of that task. Regardless of the task or the artifact, its interpretation, context, or the way users use the artifact, the space inside the magic circle stays there.

Another way to consider this is that all of the tasks that have artifacts devoted to them have essentially bounded an area, seeded it, and will keep reinforcing the boundaries. The number of tasks has ever-increased and within the operating environment we now have boundless, single-task-devoted applications. While there are an infinity of tasks that can be performed, these artifacts cannot communicate with one another save for occasional handshakes or token exchanges.

And what's more interesting is that all of these applications are created for tasks that require data connections, electricity, and infrastructure. Artifacts exist in reference to tasks and the underlying assumption of both is that there is a stable environment to perform that task in.

Carroll and Rosen [13] outlined a way to extend this task-artifact cycle through "scenario-based design." This action-oriented method allows for designers to escape the task and make cases for certain kinds of use that go beyond the task itself. While useful for extending tasks and artifacts to potentially overlap one another, there are 3 issues with this.

First, the claims of these scenarios unfortunately often only focus on what is good or what will minimize controversy and increase user-retention. For example, "the mulching of the elderly" outlines the issues surrounding this approach in that fairness, accountability, and transparency are not inherently good [35]. Second, the dissonance of what a designer intends and what a user demands is almost always out-of-balance [11, 71]. Therefore, attempting to guess what users want through scenarios is a double-edged sword and both edges could cut users off from their prior knowledge and the task itself.

Finally, the task-artifact cycle has fostered a new type of issue given the growing dissonance between designers and users. There is now a vast ecosystem of applications that cannot communicate with each other or be used for anything but what they are intended to be used for [39]. As such, the dissonance of the sum of all artifacts and their tasks are a tightly compacted, exhaustible list of tasks for stable environments. Use when an environment is not stable means that the space of use for that artifact cannot form at all, use cannot occur. While future attempts would be made to overcome these issues, approaches like adaptive design did not achieve their intended effect [23, 46]. We suggest that play is a more useful heuristic for re-considering design.

2.3 Making the Task-Artifact Cycle Extensible

To make the task-artifact cycle [11] extensible, we suggest that we need to think about design through the lens of play. With regard to play; however, it is important to first fully define what play has and currently meant. In the past, play has been defined by Huizinga as freedom independent of real life that contains its own source of order and goals [32]. While useful, Huizinga wrote this definition as modernity and empiricism swept throughout Europe and the United States just before World War 2 [32]. Later, Callois [9] added to Huizinga that play always began with uncertainty and was guided by rules and contained imaginaries that may or may not be set against the real world. More recently, Salen and Zimmerman [64] condensed these definitions to a more simplistic but direct call for play as freedom within constraint.

For the purposes of the present research, we define *play* as an act that does not explicitly identify its affordances (or the potential abilities of an object [52]) before the act begins [7]. Within that act, expression is limited by the focus of the player and limitations of the space in which the act is performed. Play is not a diversion, but a name for the act of *getting something to work* and observing

the results [7]. This definition of play allows researchers to identify a number of aspects around which to design research.

Within this definition of play, space and place are still created, it is where the action is [26]. This space, sometimes called a magic circle [32] or playground [7] is impermanent which is why it fits the task-artifact cycle so well. It is constantly reified, dissolved, reformed, and is recursive in that one could go to work and within work, do something new but all circles will dissolve in roughly similar orders. Within those circles, an order exists that is dependent on the people and things inside of it. That order is the task-artifact cycle. The drift between a task and an artifact speaks to the power of constraints embedded in affordances but also to the impermanence of tasks.

By assigning and defining play in this way, the task becomes both focal and extended. Focal in that affordances are still the primary point of design but extended in that affordances are not designed for the task in mind, but their potential for other things in the same space. Disagreement, workarounds, and appropriating affordances for other means are all potentialities within both this definition and the spaces of getting something to work. Play loosens tasks by essentially affording for appropriation potential by allowing the imaginary to exist in the design process. Through this, the task-artifact becomes extensible, we can add tasks to an artifact that are tangential to its central task. To demonstrate this approach, we describe a fictional product we will refer to as “The Decentralized Layer.”

3 The Decentralized Layer

This section describes the product in general. Our approach, described above, presumes that the sheer number of devices and apps provides an interesting way to think about safety, accessibility, danger, and destruction of local environments. In essence, we can design software using software already designed and installed. Each city, municipality, neighborhood, or area is connected to a centralized network. That layer of connectivity can be disrupted by everything from car accidents to hurricanes. When this occurs, that area ceases to be able to communicate with any other portion of the centralized network. Yet, while inter-connectivity may be missing, residents of the city will still have objects like phones, tablets, laptops, and portable gaming devices. We envision a decentralized layer, one that allows residents in an area suddenly unable to access the internet, to use their devices to chat with each other and alert first responders to their location and their situation.

In order for this type of system to work, it must be able to exist outside of the centralized web and without connection to the power grid. At the user level, basic functionality must be in place, basic in a sense that it uses little battery power. The functionality needed for this design should have the following parameters.

The system needs a way:

1. to connect devices (hardware);
2. for devices to communicate within that connection (software);

3. to locate the device (and, by proxy, the owner) and provide location data;
4. to provide some context about the user's situation at their location;
5. to note if the user is with others;
6. to note if the user is injured or impaired;
7. to communicate with those around them;
8. to remain private from others but not emergency management; and
9. to call those nearby due to an emergency.

This system encapsulates a number of needs within emergency situations. It also allows for devices that cannot connect to anything to gain functionality as a device that can be used within the current vicinity. All that is required is a Bluetooth-based connectivity hub that can host a mesh network. The software required is an application that can capture location-based data in order to feed to search and rescue (SAR) team members. The final portion is a way for that data to get to emergency management.

3.1 The Hardware

The hardware required for this service must be installed and maintained by local or city government. Private contractors often require terms of service that can be aimed at data sharing or data gathering. So by installing and maintaining these themselves, it can remain open and limited to local use much like tornado sirens or other kinds of emergency infrastructure. But much like tornado sirens, the device must be straightforward, incorporate existing infrastructure, and be invisible, but regularly visible.

To this then, the objects must:

1. have their own rechargeable battery power that is triggered by grid failure;
2. be able to survive catastrophic conditions (e.g., earthquake, hurricane, fire, gun shots, grenades, flooding);
3. be able to last on battery power for up to 24 h;
4. be robust to operating system, hardware, and standard changes;
5. be able to broadcast their signals up to a mile and mesh with other objects of the same type; and
6. (where absolutely necessary) be able to be accessed by a variety of private industry partners by an open-source add-on.

Physically, the box is perhaps the easiest part. The box can be made of cast iron or a similar metal. This would allow the box itself to sustain heavy damage and remain intact. Additionally, these boxes could be placed on the sides of buildings like parking garages, placed on the sides of cell phone towers, and used as park bench infrastructure if they were made longer. Embedded within these boxes could be filaments or wiring that would use the metal as an antennae or signal booster. What is inside the box is more important as while the box itself may be able to sustain heavy damage, the equipment within must be able to survive as well and that is far more delicate.

At their core, the box contains a Bluetooth transceiver connected to a hard drive though more work is needed to explore requirements in terms of potential additional hazards related to batteries. This additional work is needed because Bluetooth transceivers are delicate devices that cannot sustain things like water, impact, or surges. Therefore, the box must be filled with some sort of foam or non-conductive rubber-like material that can allow the objects to survive destructive events. For additional survivability, the transceiver and hard drive can be stored inside of a small enclosure like an Arduino or 3D-printed open-source enclosure. Each device has minimal software on it and has enough space to store the text-based data that will be generated during its use.

Each Bluetooth transceiver is part of a wide-scale mesh network that can be accessed by a variety of applications using an open-source product [2,60]. Functionality is also included at the administrator side for drone operators to connect to the Bluetooth network in order to download and send the data from the hard-drives on the network. The network can be activated for testing using a routine created during its installation. Our example application focuses heavily on play and the mode of exploration that is situated between the two: the map and its interface.

3.2 The App in Specific

In this section, we outline the type of software that will be used to access that network. LBGs have been pushing players out into the city since their creation [44,50,55]. Games like *Pokémon GO*, *Harry Potter: Wizard's Unite*, and *Ingress*, allow play to be mixed with exploration [27,55,56]. The use of maps as a source of play is well-documented [67,68] and, while LBGs provide a rich stream of research for those interested in maps (e.g. [3,27,31,33,57]), they could potentially serve another use [58].

As a game situated with a map, it is constantly downloading user locations for the local client and storing that data server side. Users who have these types of games on their phone are used to the type of affordances and scenarios that encompass use of these games. The combination of play-based exploration via a map provides an ample affordance space to players whose locations may suddenly become of supreme importance post-catastrophe.

Therefore, we propose that it is possible to use the structure of these games as a beacon (e.g. [42]) or as a client on a mesh network. When the phone cannot locate a cell phone signal, users can instead search for Bluetooth-based access points. Once on that Bluetooth access point, users can load one of these games that have installed the crisis client on it in order to load the crisis interface. See Fig. 1 for a graphic representation of this system in action.

Overall, the app can be deployed on any software that a) has allowed the decentralized layer to tie-in to portions of its environment and b) has a dedicated number of developers maintaining its compatibility with different versions of that software. And that software that ties in to existing software is stripped down to maximize battery time. The application operates from end-to-end in the following three ways.

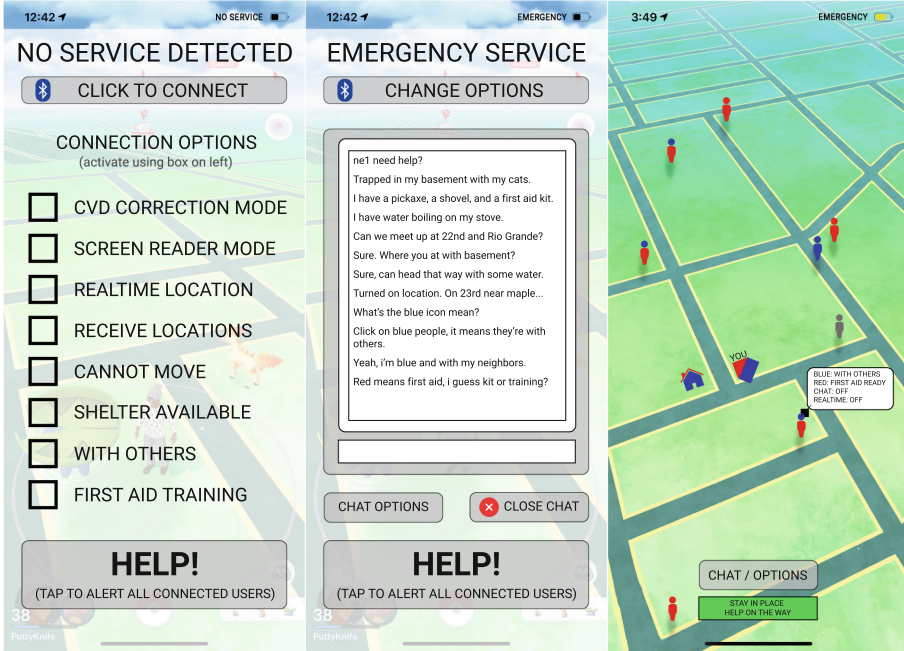


Fig. 1. In this series of illustrations, we can see a user (left) being prompted with the discovery that no service is detected on the device and that airplane mode is not activated. The user is offered a series of options to connect with as well as a button to press that will begin a search for Bluetooth-based connections. In the next illustration (center), the user has connected to the service via Bluetooth. The user can see a chat room and has basic functionality like send to chat, send a ping out broadcasting their location, and to close the chat. Finally, on the right, the user can see the current (last known) position of users on the map itself as represented by the *Pokémon GO* user-interface. On this screen, different colors on different body parts represent those users having selected different options. Below the button to re-open chat is a status indicator that will allow users to know if a drone has connected to the signal and downloaded location data or not. Once it has, the drone will send that data back to base camp to communicate to Search and Rescue (SAR) personnel. (Color figure online)

First, when users are in a crisis situation, the applications that can access the decentralized layer can be launched and, once they detect no signal, be prompted to connect. The connection between the mesh network and the devices will first, store location data inside of the hard drive for each node active on the network. By storing locations, with time stamps and additional options selected by the users, drones flying to take initial images of the impacted area will connect to the network and download a delimited data file. There is little need to store images or anything more than basic data that can be sent and evaluated quickly in order to understand where survivors may be. While this focuses on the parameters of the

initial connection, there are a number of additional options for that connection before it begins.

Next, as users engage the decentralized layer, they are prompted with the following list of options.

- **CVD (Colorblind) Correction Mode** - This ensures that the colors used by the device will be corrected to maximize the color scheme for other options.
- **Screen Reader Mode** - The data for this option will download a packet from the mesh network that will allow the device to read map-based data.
- **Realtime Location** - This will allow the app to maintain a realtime location triangulated by the nearest Bluetooth Nodes. While this will additionally maintain location data for this device, it will constantly delete and replace the line associated with the device in the delimited data.
- **Receive Locations** - This option will activate realtime location and will download data from the mesh network in order to populate a realtime map on top of the existing interface of whatever app is used to access the decentralized layer.
- **Cannot Move** - When this option is marked, it shows the search and rescue team that is evaluating the data that this person is potentially pinned down or otherwise trapped in rubble. If this item is selected, it will automatically populate a location for other people on the mesh network.
- **Shelter Available** - This option notes on the map that you have room to accept people who might have lost their shelter. This option can be turned off any time in order to show that no more shelter space is available.
- **With Others** - So many people are often surrounded by co-workers, family members, or other members of the public. Marking this option will activate realtime location but show a crowd on screen.
- **First Aid Training** - This option allows members of the mesh network to designate themselves as having received some sort of first-aid training.

Each of these options, can – as with most things – be abused by spoil sports [19,32]. Abuse related to knowing the location of survivors is not without a need for caution. This abuse potential includes marking one's self as having room to shelter others, being around others, and marking one's self as trapped or unable to move. None of these options have to be checked. Simply accessing the network allows the SAR team to know that you are there. Additionally, connecting to the network in any way will allow you to connect to chat.

The chat function is a replica of a localized internet-relay chat (IRC) server wherein chat requires few centralized resources. Everything about this app – save use of Bluetooth power (though this is changing thanks to Bluetooth Low Power [6,75] – is meant to keep phones using as little charge as possible. Chat though, is available for those who need it though it too can be abused. However, given that these applications are not meant to be used unless the area around the user is damaged or otherwise in crisis, this level of risk is reasonable. Training and other forms of help-based text can help users understand risk though we would expect most users to be more concerned with their survival at the time

of need. How to overcome this issue requires the paradox of the active user to be confronted [12].

Third and finally, the map interface itself allows users to witness who is near them. Each device is shown as a low-poly figure that is color-coded according to the options that they picked. While viewing the map, each individual that is nearby can be touched to see what options they have selected. When the option for having shelter available is marked, it will be shown as a small house on the map. By showing who is where and what options have been selected, users can congregate, or avoid, others. Or, if users are trapped and unable to move, they may be able to be located and helped before SAR teams arrive.

This aspect of the map will be draining on batteries. As such, it will be updated every 30 s. At the bottom of the map where users can go back to options and the chat box will be a color-coded box that marks 3 different options:

- **Red** - This color means that that device’s data has not been downloaded by a drone yet. This color will include the message, “location not delivered.”
- **Yellow** - This color means that the device’s data has been downloaded but that nothing has been dispatched to the field quite yet. This color will include the message, “location delivered, awaiting dispatch.”
- **Green** - This color notes that the users in that area should stay in place and that help is on the way. This color will include the message, “Stay in place. Help on the way.”
- Each of these colors can be altered for color blindness.

The way that this data is updated is first by a drone accessing the data and transmitting it to base camp. The data is then analyzed and a special package is sent to another drone that is monitoring traffic. People in a geographic location that has been assigned to a SAR team will be sent a special ping that changes the color and message on that specific bar.

4 Author’s Statement and Discussion

Within the current mode of design, production, and deployment, the task-artifact cycle stands as the standard mode of operations. While this mode has been useful and productive, the likelihood that software devoted to a task will include tie-ins for additional tasks or potentials is extremely low. However, there is an opportunity (and potentially a requirement) for extensible design to afford these software far more uses than they currently possess and these can be pursued with, or without the original creators. We have presented a play-centric interpretation of the task-artifact cycle we call extensible design. Within this paradigm, we presented a fictional approach to design, one that takes existing products and blends portions of the artifact in a way as to allow them to be re-purposed when needed.

The application we outlined in the previous design fiction presents what we refer to as the “decentralized layer,” a bounded space that will appear and

connect internet-enabled devices temporarily given the eventuality of a disaster-level event. We outlined this layer and an application to show what the extended task-artifact cycle would perceive of existing tasks, potential needs within the city, and how to begin to design around them. While disruptive in its own right in terms of intellectual property, private or closed source software, and issues involving permissions and rights on operating systems, what this design provides is a fictional account of a way that designers could maintain their normal mode of operation while fostering a more ingredient-based aspect of design.

An important aspect of this design is not so much the basics of how it could work should it exist, but in the infrastructure and additional work needed for the application to work at all. The assumptions made by the authors allows us to re-contextualize and explain some aspects of the design that have to either be answered elsewhere or worked around. Next, because of the way that software has been created over time it is important to note that there is a supreme amount of labor, hardware, and materials that need to be paid for by someone. Finally, there are the users themselves. How do users 1) learn that these systems exist 2) download them before a disaster event and 3) use them under potentially life-threatening conditions?

4.1 The Assumptions Made by the Authors

The assumptions of this design are set by the authors in order to highlight the needs of a product that is designed to support disruption in addition to its normal functions. In this case, the design is focused on large-scale disasters and incorporate software, especially software that use maps. It should be this way because this type of design will provide essential services according to the author's previous work in non-GIS-based maps and mapping technologies [38, 67, 68]. Further, in the interest of keeping this design fiction as non-fiction as possible, we needed to use products that exist and materials that exist.

Since we are designing for a municipal government and are partnering with private industry (in this case, Niantic, makers of *Pokémon GO*), there are additional aspects specific to what we have in mind. First, while private industry will have the capacity to absorb certain kinds of expenses at the behest of government entities, local governments will have to pay for hardware themselves. As a result, there are three specific assumptions that center on hardware and the city's perspective.

1. From material to components, everything must be off-the-shelf, easily installed, inexpensive, and physically hearty.
2. Everything must be decentralized but use existing technologies that will remain compatible for a time of 10 years or more.
3. Everything must be testable (like tornado sirens), public, and open.

By keeping hardware to easily used and configured off-the-shelf components, these can be purchased in bulk and assembled quickly. Alternatively, an open-source community could offer municipalities access to their systems at cost [30].

An aspect of designs like these will be in keeping up with operating system patches, application patches, and other ways that hardware and software lose compatibility. As a result, these products must be selected on the basis of their continual use. Finally, testing is incredibly important for any safety-oriented tool. Much like tornado sirens, software like this needs to have a test mode that activates the network. This can be a tremendous source of city participation with mesh-network-based messages to all participants on the network for password protected deals and brief sales.

In following these parameters, we can appropriately ground any hardware design that comes out of this extended version of the task-artifact cycle. Next, in order to ground this even further, we needed another metaphor to help situate this product in terminology that designers are used to. This will help us move from describing this work through play, a somewhat opaque and loaded term, to something more useful for designers. The use of a metaphor will guide us toward the intent behind the product we will describe. In doing this, not only can we deploy extensible design, but provide new tools for consumers to use in the midst of disaster that emergency management can use easily, thus meeting the suggestions of crisis informatics about technology adoption [54,62].

We used the concept of “extensible” languages in computer science like lua or, at least, extensible grammars like Perl6, Lisp, Red, and Racket to achieve this end [70]. By extensible, we mean that, within these languages, it is possible to move beyond the limitations of a controlled environment. Programmers can instead create new types of structures or syntax that extend the operation of these products to places that the creators and maintainers of the languages did not expect. This metaphor is appropriate as the previous sections outlined a design paradigm centered on play and disruption. Design would – by default – be outside the intended limits of the environment.

By extending the design of products like that of extensible languages, we can conceptualize hardware that can be accessed by many different programs with specific, open and pre-installed software. With software, the competitive, profit-driven aspect of the apps aimed at extending is as such that there needs to be a set of limits for extension in order to minimize their diversity. And so, within the existing three assumptions that center on the environment that the hardware will exist in, we then see three additional assumptions for the software portion of this design, or the private industry’s involvement. These assumptions are:

1. Installed with existing applications.
2. Easily applied to any design or language, open source, and well-commented.
3. Applications that use these hardware will be advertised for free throughout the city.

With this in mind, the components of this design are hardware and software in the form of an application that is installed on the side of existing applications. Since this product would exist within the city and is safety-oriented, the type and severity of the emergency being used as a design inspiration is important. This

would allow cities to offer posters and training to use these products. Publicity can also be performed through private partners who open up their products to those interested in the initiative.

In this capacity, we will designed around total failure states. For example, a flood, widespread destruction due to an earthquake, or failure of the electrical grid of a particular area of a city due to some sort of event. While this type of event is rare, it allows us to highlight certain kinds of potential for extensible design. Through these assumptions, we are able to 1) reach designers using terminology that they can understand, 2) grounded design in the nearest future possible, and 3) set up a context for use and design. In the next section, we further discuss the maintenance and installation of those systems.

4.2 Software Creation, Maintenance, and Change

Extensible design is perhaps a thought manifested by a pollyanna, those who see things only as positive and ignore all negatives. Extensible design, as outlined, may potentially neglect the incredibly complicated politics and history of software production, city management, civil engineering, disaster science, and innumerable other domains. Yet, we believe that extensible design affords each of those domains a new approach to consider, one that may potentially get us away from the stalemates and stopping points of those domains. By maneuvering focus away from artifacts created in the current task-artifact cycle, extensible design can foster more consideration for the social life of apps after their creation and those outside of the task-artifact cycle.

Cities and municipalities do not often have the capacity to maintain, produce, or install this type of product. Civic software has often been seen as a useless gesture or at best, one meant to provide support from a company that is then given to a municipality and summarily abandoned due to its lack of support [69]. By fostering cooperation among different actors within the creation process, maintenance, change, and creation is distributed to the effect that one would expect the Pareto principle, or power laws, to carry the creation, maintenance, and change forward by removing a single driver from the project [74].

The software created for this applications is created and maintained by open source allies. This space has recently begun to maneuver itself into becoming more welcome when new contributors engage a product [43]. Additionally, through concepts like the software carpentry movement [72, 73], participation in highly technical aspects of programming is becoming more widespread. By fostering these aspects of design, of democratizing design, this method of mutually dependent development can foster further cooperation among disparate actors. Open source work with corporate contributors has become a far more pronounced aspect of open source work and should provide much needed lubricant for acceleration in this space [34].

4.3 User Training and Knowledge Building

The last aspect of this product is one of the paradox of the active user [12]. Within this concept, users will bring their knowledge of how other tasks have worked in the past through artifacts and attempt to deploy that knowledge on the product before them. The crux of this paradox is situated within the task-artifact cycle. The potential to misinterpret a task when it is translated by a design team and embedded in an artifact results constantly rises. The likelihood of disagreement between various interpretations of that task and because software design is hierarchically situated, designers will always have power of that artifact and, by and large, the task itself. By extending the cycle through play, by understanding that a product should be situated not in an experience, not in a task, but as an ingredient that can be used in multiple ways, we expect that the paradox of the active user could not only be ignored, but overcome.

This is perhaps the most difficult aspect of this design fiction to accept as suspension of disbelief can only be pushed so far. Users using products how they want, or how they need to independent of user experience design, user interface design, or a product's overall afforded intent is simply removed in favor of use independent of design. Yet, much like *Robinson Crusoe* in *Friday, the Other Island*, detonating all of those concepts may irrevocably free us [41] from the task-artifact cycle completely. As such, there is little to say here but to "wait and see."

5 Conclusion

In this design fiction, we outlined "extensible design" or the use of existing software in special circumstances. We defined extensible design by situating play, the constant formation of self-contained, self-defining spaces, as a lens through which to see design as ingredient-focused rather than task-artifact-focused. We then provided a fictional description of a potential system that could be built with this lens. The system, the "decentralized layer" is a 2-part system that places Bluetooth transceivers throughout a city that will activate given the catastrophic loss of power.

Once the system activates, users can access a decentralized network that additionally uses existing software to connect users to chat, location-based affordances, and brief signaling options that can be easily downloaded and sent back to a SAR basecamp so that personnel can be dispatched to areas where help is needed most. While waiting for help, users within the decentralized layer can find each other, self-organize, and triage each other's harm as best they can before trained help arrives. While this application requires a moderate amount of work dissolving barriers of access, barriers of resources, and barriers of computational knowledge, this lens provides a way through which to 1) view those barriers, 2) understand how they are created, and 3) distribute responsibility to enough partners to allow the project to succeed.

More design fiction, more world building, is needed to realize the potential of extensible design. The sheer number of mobile devices and applications installed

in the centralized network that connects us all to each other has other uses and this concept of extensible design affords those objects more uses than they currently contain. Play, though it may seem antithetical, is an essential lens for that confrontation. Through the lens of play, we see apps and devices not as a magic circle in and of themselves, but as another ingredient, another tool through which to consider design. And in so doing, our obscenely complex ecosystem of apps, devices, and users can find new avenues of growth.

References

1. Aarseth, E.: Computer game studies, year one. *Game studies* **1**(1), 1–15 (2001)
2. Aggelou, G.: *Wireless Mesh Networking*. McGraw-Hill Professional, New York (2008)
3. Alha, K., Koskinen, E., Paavilainen, J., Hamari, J.: Why do people play location-based augmented reality games: a study on Pokémon GO. *Comput. Hum. Behav.* **93**, 114–122 (2019)
4. Backus, J.W., et al.: Report on the algorithmic language ALGOL 60. *Commun. ACM* **3**(5), 299–314 (1960)
5. Baumer, E.P., Blythe, M., Tanenbaum, T.J.: Evaluating design fiction: the right tool for the job. In: *Proceedings of the 2020 ACM Designing Interactive Systems Conference*, pp. 1901–1913 (2020)
6. Bernardos, A.M., Bergesio, L., Metola, E., Ortiz, D., Casar, J.R.: Deploying a BTLE positioning system: practical issues on calibration. In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6. IEEE (2017)
7. Bogost, I.: *Play Anything: The Pleasure of Limits, the Uses of Boredom, & the Secret of Games*. Basic Books, New York (2016)
8. Boom, K.H., Ariese, C.E., van den Hout, B., Mol, A.A., Politopoulos, A.: Teaching through play: using video games as a platform to teach about the past. In: *Communicating the Past*, vol. 27 (2020)
9. Caillois, R.: *Man, Play, and Games*. University of Illinois Press, Champaign (2001)
10. Card, S.K., Moran, T.P., Newell, A.: The model human processor - an engineering model of human performance. *Handb. Percept. Hum. Perform.* **2**(45–1) (1986)
11. Carroll, J.M., Kellogg, W.A., Rosson, M.B.: The task-artifact cycle. In: *Designing Interaction: Psychology at the Human-Computer Interface*, pp. 74–102. Cambridge University Press (1991)
12. Carroll, J.M., Rosson, M.B.: Paradox of the active user. In: *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction*, pp. 80–111. MIT Press (1987)
13. Carroll, J.M., Rosson, M.B.: Getting around the task-artifact cycle: how to make claims and design by scenario. *ACM Trans. Inf. Syst. (TOIS)* **10**(2), 181–212 (1992)
14. Corsar, D., Edwards, P., Baillie, C., Markovic, M., Papangelis, K., Nelson, J.: Get-There: a rural passenger information system utilising linked data & citizen sensing. In: *Proceedings of the 12th International Semantic Web Conference (Posters & Demonstrations Track), ISWC-PD 2013*, vol. 1035, pp. 85–88. CEUR-WS.org, Aachen, DEU (2013)
15. Coulton, P., Lindley, J., Sturdee, M., Stead, M.: Design fiction as world building. In: *Proceedings of the 3rd Biennial Research Through Design Conference*, pp. 163–179 (2017)

16. Daylight, E.G.: Dijkstra's rallying cry for generalization: the advent of the recursive procedure, late 1950s-early 1960s. *Comput. J.* **54**(11), 1756–1772 (2011)
17. Daylight, E.G., Grave, K.D.: *Pluralism in Software Engineering: Turing Award Winner Peter Naur Explains. Lonely Scholar*, Zurich (2011)
18. Daylight, E.G., Wirth, N., Hoare, T., Liskov, B., Naur, P., Grave, K.D.: *The Dawn of Software Engineering: From Turing to Dijkstra. Lonely Scholar*, Zurich (2012)
19. DeKoven, B.: *The Well-Played Game: A Player's Philosophy*. MIT Press, Cambridge (2013)
20. Deterding, S.: The pyrrhic victory of game studies: assessing the past, present, and future of interdisciplinary game research. *Games Culture* **12**(6), 521–543 (2017)
21. Dijkstra, E.W.: Recursive programming. *Numer. Math.* **2**(1), 312–318 (1960)
22. Dijkstra, E.W., et al.: *Notes on structured programming* (1970)
23. Dix, A.: Designing for appropriation. In: *Proceedings of HCI 2007 The 21st British HCI Group Annual Conference University of Lancaster, UK*, vol. 21, pp. 1–4 (2007)
24. Dix, A.: Human-computer interaction, foundations and new paradigms. *J. Vis. Lang. Comput.* **42**, 122–134 (2017)
25. Dourish, P.: The appropriation of interactive technologies: some lessons from placeless documents. *Comput. Supported Cooper. Work (CSCW)* **12**(4), 465–490 (2003)
26. Dourish, P.: *Where the Action Is: The Foundations of Embodied Interaction*. MIT Press, Cambridge (2004)
27. Dunham, J., Papangelis, K., LaLone, N., Wang, Y.: Casual and hardcore player traits and gratifications of Pokémon GO, Harry Potter: Wizards Unite, Ingress. *arXiv preprint arXiv:2103.00037* (2021)
28. Dunne, A., Raby, F.: *Speculative Everything: Design, Fiction, and Social Dreaming*. MIT Press, Cambridge (2013)
29. Galloway, A., Brucker-Cohen, J., Gaye, L., Goodman, E., Hill, D.: Design for hackability. In: *DIS 2004: Proceedings of the 5th Conference on Designing Interactive Systems*, pp. 363–366. ACM Press (2004)
30. Germonprez, M., Hovorka, D.S.: Member engagement within digitally enabled social network communities: new methodological considerations. *Inf. Syst. J.* **23**(6), 525–549 (2013)
31. Hamari, J., Malik, A., Koski, J., Johri, A.: Uses and gratifications of Pokémon GO: why do people play mobile location-based augmented reality games? *Int. J. Hum.-Comput. Interact.* **35**(9), 804–819 (2019)
32. Huizinga, J.: *Homo Ludens: A Study of the Play-Element in Culture*. Martino Fine Books, Eastford (2014)
33. Jones, C., Papangelis, K.: Reflective practice: lessons learnt by using board games as a design tool for location-based games. In: Kyriakidis, P., Hadjimitsis, D., Skarlatos, D., Mansourian, A. (eds.) *AGILE 2019. LNGC*, pp. 291–307. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-14745-7_16
34. Kendall, K.E., Kendall, J.E., Germonprez, M., Mathiassen, L.: The third design space: a postcolonial perspective on corporate engagement with open source software communities. *Inf. Syst. J.* **30**(2), 369–402 (2020)
35. Keyes, O., Hutson, J., Durbin, M.: A mulching proposal: analysing and improving an algorithmic system for turning the elderly into high-nutrient slurry. In: *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–11 (2019)
36. Knutz, E., Markussen, T.: The role of fiction in experiments within design, art & architecture-towards a new typology of design fiction. *Artifact: J. Des. Pract.* **3**(2), 8–1 (2014)

37. Kruseman Aretz, F.: The Dijkstra-Zonneveld ALGOL 60 compiler for the electrologica X1 (2003)
38. LaLone, N., Alharthi, S., Toupes Dugas, P.O.: A vision of augmented reality for urban search and rescue. In: Proceedings of the 1st Halfway to the Future Symposium, pp. 1–5. ACM, Nottingham, UK (2019)
39. LaLone, N.J.: Association mapping: social network analysis with humans and non-humans. Ph.D. thesis, The Pennsylvania State University (2018)
40. Lambrow, A.: The seriousness of play: Johan Huizinga and Carl Schmitt on play and the political. *Games Culture* **16**(7), 820–834 (2021)
41. Latour, B.: The Pasteurization of France. Harvard University Press, Cambridge (1993)
42. Lin, X.Y., Ho, T.W., Fang, C.C., Yen, Z.S., Yang, B.J., Lai, F.: A mobile indoor positioning system based on iBeacon technology. In: 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 4970–4973. IEEE (2015)
43. Lombard, K., Buhman, A., Wethor, G.E., Hale, M., Goggins, S., Germonprez, M.: Welcome? Investigating the reception of new contributors to organizational-communal open source software projects. In: AMCIS 2020 Proceedings. AMCIS (2020)
44. Matyas, S., Matyas, C., Schlieder, C., Kiefer, P., Mitarai, H., Kamata, M.: Designing location-based mobile games with a purpose: collecting geospatial data with CityExplorer. In: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology, pp. 244–247 (2008)
45. Meo, R., Roglia, E., Bottino, A.: The exploitation of data from remote and human sensors for environment monitoring in the SMAT project. *Sensors* **12**(12), 17504–17535 (2012)
46. Moran, T.P.: Everyday adaptive design. In: Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, pp. 13–14 (2002)
47. Muratori, C.: Handmade hero (2021). <https://handmadehero.org/>
48. Naur, P.: Programming as theory building. *Microprocess. Microprogram.* **15**(5), 253–261 (1985)
49. Naur, P.: Computing: a human activity. ACM (1992)
50. Nicklas, D., Pfisterer, C., Mitschang, B.: Towards location-based games. In: Proceedings of the International Conference on Applications and Development of Computer Games in the 21st Century: ADCOG, vol. 21, pp. 61–67 (2001)
51. Noortman, R., Schulte, B.F., Marshall, P., Bakker, S., Cox, A.L.: HawkEye - deploying a design fiction probe. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, pp. 422:1–422:14. ACM, New York (2019). <https://doi.org/10.1145/3290605.3300652>
52. Norman, D.: The Design of Everyday Things: Revised and Expanded Edition. Basic Books, New York (2013)
53. Palen, L., et al.: Crisis informatics: human-centered research on tech & crises (2020)
54. Palen, L., Anderson, K.M.: Crisis informatics-new data for extraordinary times. *Science* **353**(6296), 224–225 (2016)
55. Papangelis, K., Chamberlain, A., LaLone, N., Cao, T.: Insights and lessons learned from the design, development and deployment of pervasive location-based mobile systems “in the wild”. In: Soares, M.M., Rosenzweig, E., Marcus, A. (eds.) HCII 2021. LNCS, vol. 12781, pp. 79–89. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-78227-6_7

56. Papangelis, K., et al.: Locating identities in time: an examination of the formation and impact of temporality on presentations of the self through location-based social networks. *ACM Trans. Soc. Comput. (TSC)* **4**(3), 1–23 (2021)
57. Papangelis, K., Metzger, M., Sheng, Y., Liang, H.N., Chamberlain, A., Khan, V.J.: “Get off my lawn!”: starting to understand territoriality in location based mobile games. In: *CHI EA 2017 Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 1955–1961. ACM Press (2017). <http://dl.acm.org/citation.cfm?doid=3027063.3053154>
58. Papangelis, K., Sheng, Y., Liang, H.N., Chamberlain, A., Khan, V.J., Cao, T.: Unfolding the interplay of self-identity and expressions of territoriality in location-based social networks. In: *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*. ACM (2017). <https://doi.org/10.1145/3123024.3123081>
59. Pellicer, J.L., Blanes, J.S., Tormos, P.M., Frau, D.C.: Using processing.org in an introductory computer graphics course. In: *Eurographics 2009-Education Papers*, pp. 23–28 (2009)
60. Raniwala, A., Chiu, T.C.: Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. In: *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 2223–2234. IEEE (2005)
61. Reeves, S., Goulden, M., Dingwall, R.: The future as a design problem. *Des. Issues* **32**(3), 6–17 (2016)
62. Reuter, C., Hughes, A.L., Kaufhold, M.A.: Social media in crisis management: an evaluation and analysis of crisis informatics research. *Int. J. Hum.-Comput. Interact.* **34**(4), 280–294 (2018)
63. Reuter, C., Kaufhold, M.A.: Fifteen years of social media in emergencies: a retrospective review and future directions for crisis informatics. *J. Contingencies Crisis Manag.* **26**(1), 41–57 (2018)
64. Salen, K., Zimmerman, E.: *Rules of Play: Game Design Fundamentals*. MIT Press, Cambridge (2004)
65. Tan, M.L., Prasanna, R., Stock, K., Hudson-Doyle, E., Leonard, G., Johnston, D.: Mobile applications in crisis informatics literature: a systematic review. *Int. J. Disaster Risk Reduction* **24**, 297–311 (2017)
66. Toftedahl, M., Engström, H.: A taxonomy of game engines and the tools that drive the industry. In: *DiGRA 2019, The 12th Digital Games Research Association Conference*, Kyoto, Japan, 6–10 August 2019. Digital Games Research Association (DiGRA) (2019)
67. Touns Dugas, P.O., Lalone, N., Alharthi, S.A., Sharma, H.N., Webb, A.M.: Making maps available for play: analyzing the design of game cartography interfaces. *ACM Trans. Comput.-Hum. Interact.* **26**(5), 30:1–30:43 (2019). <https://doi.org/10.1145/3336144>
68. Touns Dugas, P.O., LaLone, N., Spiel, K., Hamilton, B.: Paper to pixels: a chronicle of map interfaces in games. In: *Proceedings of the 2020 ACM Designing Interactive Systems Conference, DIS 2020*, pp. 1433–1451. Association for Computing Machinery, New York, NY, USA (2020). <https://doi.org/10.1145/3357236.3395502>
69. Townsend, A.M.: *Smart Cities: Big Data, Civic Hackers, and the Quest for a New Utopia*. WW Norton & Company, New York (2013)

70. Tsai, W.T., Tu, Y., Shao, W., Ebner, E.: Testing extensible design patterns in object-oriented frameworks through scenario templates. In: Proceedings of the Twenty-Third Annual International Computer Software and Applications Conference (Cat. No. 99CB37032), pp. 166–171. IEEE (1999)
71. Vermeulen, J., Luyten, K., van den Hoven, E., Coninx, K.: Crossing the bridge over Norman’s Gulf of Execution: revealing feedforward’s true identity. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1931–1940 (2013)
72. Wilson, G.: Software carpentry: getting scientists to write better code by making them more productive. *Comput. Sci. Eng.* **8**(6), 66–69 (2006)
73. Wilson, G.: Software carpentry: lessons learned. *F1000Research* **3** (2014)
74. Yamashita, K., McIntosh, S., Kamei, Y., Hassan, A.E., Ubayashi, N.: Revisiting the applicability of the pareto principle to core development teams in open source software projects. In: Proceedings of the 14th International Workshop on Principles of Software Evolution, pp. 46–55 (2015)
75. Zhao, X., Xiao, Z., Markham, A., Trigoni, N., Ren, Y.: Does BTLE measure up against WiFi? A comparison of indoor location performance. In: European Wireless 2014; 20th European Wireless Conference, pp. 1–6. VDE (2014)