Novel Use of Scenarios in the Usability Engineering of a Next-generation MLST Tool

Stephanie M. Alpert

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Industrial and Systems Engineering

Joseph L. Gabbard, Chair
Tonya L. Smith-Jackson
Meredith J.C. Wilson

February 13, 2014
Blacksburg, Virginia

Novel Use of Scenarios in the Usability Engineering of a Next-generation MLST Tool

Stephanie M. Alpert

ABSTRACT

This work explores the utilization of scenarios in an iterative usability engineering process for the development of a next-generation multilocus sequence typing (MLST) tool. The following three research question were investigated during the usability process: (1) what are the differences in the elicited requirements as scenarios move further from extant work practices, (2) what are the differences in the elicited requirements between structured and free-form scenario groups, and (3) are participant-developed scenarios from the scenario-based interviews effective for use as tasks in formative usability evaluation.

Scenario-based interviews were conducted to collect relevant work-practice information and domain knowledge from two user classes. Requirements distilled from the scenarios and complementary interview questions informed the design of multiple iterations of the tool. A formative usability evaluation was conducted on the second iteration of the tool with the same participants.

Resulting requirements from the scenario-based interviews suggest that proposing scenarios beyond current work practices overwhelmed and confused participants, and therefore worked against requirements generation. Conversely, a less structured scenario-based interview scheme yielded a greater quantity of requirements, and specifically produced more creative requirements. Participant-developed scenarios from the scenario-based interviews were ultimately useful as benchmark tasks in the formative usability evaluation because they were intricate enough to afford meaningful interaction with the interface, while still being completable by both user classes. This research helps to provide a greater understanding of the utilization of novel scenario styles and methodologies, thereby providing support for the continued investigation into scenario use for a variety of applications.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

Scenarios are widely used for requirements elicitation in human-computer interaction (HCI) and software engineering. While there are many styles of scenarios, they traditionally focus on existing work practices. Scenarios often do not take into account new interactions and capabilities afforded by technological advances and innovation. The purpose of this research is twofold: to explore how scenarios can be used to support creative thinking about how new technologies can enhance and extend current work practices, and to facilitate creation of new scenario-based workflows to drive both requirements and subsequent usability evaluations.

This research will be done in the context of a new multi-locus sequence typing (MLST) tool being developed for the PATRIC system; an online web-based information system designed to support the biomedical research community's work on bacterial infectious diseases via integration of vital pathogen information with rich data and analysis tools. By investigating the novel use of scenarios in the context of a next-generation MLST tool, this research aims to extend the workflow of epidemiologists and molecular biologists – the two main user classes of MLST.

1.1 MLST Background

Researchers use various methods to study strains of bacteria that make people sick, in an effort to learn how to more effectively protect people from illness. Researchers study general characteristics of these harmful bacteria as they exist worldwide, but also study specific strains when there is an outbreak of an infectious disease caused by bacteria in a particular location. In outbreak cases, the goal is to identify the outbreak strain by examining a small set of signature genes, and trying to matching them to known strains of bacteria.

It is of critical importance to epidemiological surveillance and public health decision makers to accurately identify the strains of infectious agents that cause disease, especially in time-sensitive outbreak settings. Though not as time-sensitive, molecular biologists conduct research on the evolution of bacterial strains over time, and the relationships between strains discovered worldwide, which helps provide a better foundation of knowledge about potentially harmful bacteria. Many *molecular typing*

methods exist to carry out these types of research, but MLST has become known as one of the most useful and accurate.

There are various methods of molecular typing, but many of them face considerable drawbacks. One of the most critical limitations of molecular typing methods is the difficulty in comparing results achieved by different laboratories. Other significant limitations include difficulty in adequately discriminating the strain type, limited availability of materials needed to conduct the typing, and poor reproducibility (Maiden et al., 1998).

The introduction of MLST brought about many advantages as compared to previously available molecular typing methods. MLST examines the DNA sequences of a small, specific set of genes required to maintain basic cellular function. This technique affords greater resolution than other techniques due to increased information detected at each *locus*, or small portion of a gene. Another advantage is the ability to compare sequence data between laboratories, allowing strains found using equipment from one laboratory to be compared to strains in another laboratory. Additionally, this sequence data can be stored, browsed and searched via the internet. The accessibility of the multilocus sequence data through the internet provides a powerful distributed resource for standardized global epidemiology (Maiden et al., 1998).

While MLST is advantageous in many respects, the decreasing cost of whole genome sequencing opens up many possibilities to epidemiologists and molecular biologists that to date, have been unavailable. Having a whole genome sequence of an organism means possessing all of the heredity information encoded into its DNA, as opposed to just specific portions. Since MLST uses only small portions of the genes, having access to whole genome sequence data enables epidemiologists to more effectively track the transmission pathways of pathogens, and support outbreak investigations (Köser et al., 2012). Whole genome data additionally enables molecular biologists to better trace how traits travel within bacterial genealogy.

However, people accustomed to using MLST may encounter difficulty when working with whole genome sequence data as it (and its accompanying capabilities) becomes more readily available. Currently, a gap exists between the theory- and wet-lab-based research of biologists, and the technology-savvy ability of bioinformaticians to

manipulate data grounded in that theory. As such, there is a need for usable software applications that can guide users in utilizing whole genome sequence data in a familiar and intuitive way. These software applications have the potential to bridge this gap by introducing ways to extract MLST data from whole genome sequence data, as well as provide additional tools that can investigate genes at a higher resolution.

1.2 MLST and Situation Awareness

While many MLST tools are currently available, the systems can be unwieldy and unclear. Additionally, features of use to the MLST data are available, but scattered in various resources across the internet. Due to the significance of the research done using MLST, there is a need for a cohesive tool that combines relevant features, emerging capabilities inherent in whole-genome sequences, and an intuitive interface into one platform. A cohesive tool would facilitate knowledge management through the visualization of large and diverse pools of data. Situation awareness (SA) is an immediately relevant concept when considering the cohesion and intuitiveness of an interface, and should be more carefully considered when designing an MLST tool.

SA is a term that originated in aviation psychology, but has become a well-established concept in many domains such as nuclear power plant operation, air traffic control, anesthesiology, and automobile operation. The concept of SA is used broadly by researchers and practitioners to describe cognitive activities needed to control or operate a dynamic system (Durso & Gronlund, 1999). These cognitive activities may include constructs such as attention, perception, and working memory. Understanding these cognitive activities is critical to understanding decision making and performance.

1.3 PATRIC

In October 2009 – The National Institute of Allergy and Infectious Diseases (NIAID), part of the National Institutes of Health (NIH), awarded a 5-year, $27,670,448 contract to the CyberInfrastructure Division (CID) of the Virginia Bioinformatics Institute (VBI) at Virginia Tech to support the biomedical research community's work on infectious diseases. The centerpiece of this award is the design and development of the PATRIC Bacterial Bioinformatics Resource Center. Currently PATRIC is catered toward

researchers interested in comparative genomics, but aims to include resources for other relevant user classes in the future.

Incorporating a next-generation MLST tool into PATRIC will be beneficial in that it will help recruit and serve the epidemiology and the molecular biology communities. Additionally, the MLST tool will help to promote translational science by bringing biomedical research on the utilization of MLST from the lab into practice. This thesis research is being conducted in the context of the PATRIC project to promote work on bacterial infectious diseases by incorporating a next-generation MLST tool into the system. This research aims to provide a systematic human-factors based approach focusing on the novel use of scenarios to assist in designing, developing and evaluating MLST capabilities for PATRIC. Outcomes from this research will benefit PATRIC directly, but can also be leveraged in other epidemiological tools, bioinformatics web resources, and more broadly in other data intensive fields.

1.4 User-Centered Design

User-centered design (UCD) is an approach to interactive system development which takes into account human capabilities, skills, limitations, and needs to make systems usable (ISO, 1999). Gould and Lewis (1985) contributed to the beginning of the UCD movement when they advocated *understanding* representative users rather than just identifying them. They identified three design principles which help contribute to the notion of understanding representative users: early focus on users and tasks, empirical measurement, and iterative design. Norman coined the term UCD the following year, and pushed for the creation of a science of UCD principles to be applied at the time of design (Hoeft & Mentis, 2009).

UCD is now considered the key to system usefulness and usability, and an effective remedy to the limitations of traditional system-centered design (Mao, Vredenburg, Smith, & Carey, 2005). The key to the success of UCD is that the user is taken into account throughout the development process, and further throughout the entire system lifecycle. Involving the user through all stages increases the likelihood of accurate requirements, which consequently increases user satisfaction with the system. The specific UCD methodologies to be applied are tailored to meet the needs of each specific

project, but some examples of UCD methods include: interviewing representative users, card sorting, task analysis, and usability testing. This research incorporates UCD into the development of a next-generation MLST tool by utilizing interview-based elicitation methods and formative usability evaluation.

Two user classes were defined for this research: epidemiologists and molecular biologists. While epidemiologists and molecular biologists both use MLST to achieve specific work goals in their respective domains, they use them in distinct manners. Epidemiologists use MLST to identify the strains of infectious agents that cause disease. Their work with MLST can be thought of as more concrete in that they often seek out a definitive answer to a question. In other words, epidemiologists use more of a rule-based model of performance (Rasmussen, 1983) when interacting with MLST tools. Molecular biologists use MLST to conduct research on the evolution of bacterial strains over time, and the relationships between strains discovered worldwide. Their work is more indistinct in that they are generally guided by broad research questions. In other words, molecular biologists use more of a knowledge-based model of performance (Rasmussen, 1983) when interacting with MLST tools.

1.5 Scenarios

Scenario-based techniques are an example of UCD, and therefore are used to target the users' needs. Scenarios have become increasingly prevalent as a tool in fields such as software engineering and HCI. Due to their utilization across various fields they have taken on many forms, and have therefore become difficult to cohesively define. Scenarios can be thought of as a continuum from real-world stories and descriptions to specifications and models (Sutcliffe, 2003). At one end of the spectrum, a scenario is classified as a story about people and their activities (Carroll, 2000). Similar to the common notion of a story, Carroll (2000) presents characteristic elements of a scenario: an implied or explicitly stated setting, actors each with their own goals or objectives, and a plot which encompasses the sequence of actions or events taken by the actors. A scenario in this manner represents an instance or example of how a user may interact with a system. At the other end of the spectrum, a scenario is classified as a pathway through a specification of system usage. Scenarios are then aggregated into an exhaustive list,

which define all possible interactions that the user may have with the system (Sutcliffe, Maiden, Minocha, & Manuel, 1998). The example below, taken from Glinz (2000) shows two versions of the same scenario for borrowing a book from the library.

Scenario name: Borrowing books Version: Narrative Scenario

When a library users wants to borrow a book, she takes it to the checkout station. There she first scans her personal library card. Then she scans the barcode label of the book. If she has no borrowed books that are overdue and the book is not reserved for another person, the system registers the book as being borrow by her and turns off the electronic safety device of that book. Several books can be checked out together. The checkout procedure is terminated by pressing a 'Finished' key. The system produces a loan slip for the books that have been borrowed.

Scenario name: Borrowing books Version: Step-by-step
Actor: User

Normal flow
1 Scan and validate the user's library card
2 Scan book label and identify book
3 Scan label of book to be borrowed
4 Record book as borrowed, unlock safety label
5 If user wants to borrow more than one book, repeat steps 2 to 4
6 When finished, print loan slip.

Alternative flows
1.1 Card is invalid: terminate
2.1 User has overdue books: terminate
4.1 Book is reserved for another person: deny loan, continue

Figure 1. Scenario Examples (Glinz, 2000); Used under fair use, 2014

The Narrative Scenario represents the "real-world story" extreme of the spectrum presenting the scenario in a pure narrative form, as seen in Figure 2. The Step-by-step Scenario falls on the other end of the spectrum in that it explicitly states the sequences of actions that the actor takes. However, since the Step-by-step Scenario is still in narrative form it would not fall on the other extreme.

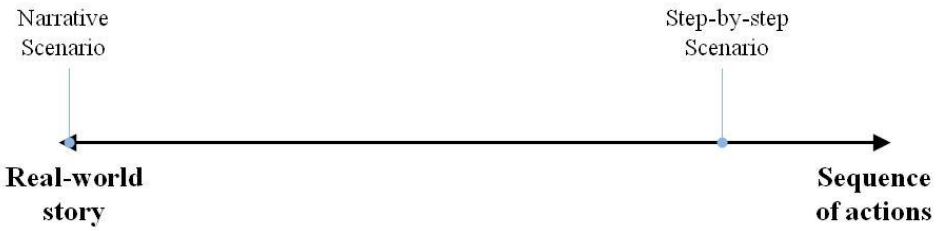


Figure 2. Spectrum of Scenario Styles

Despite the difference in presentation style of the scenario, both versions contain the characteristic elements defined by Carroll (2000). It is important to note, as Carroll (1995) wrote in the preface of the book spawned from the historic workshop sponsored by IBM, "scenario-based design is not a finished paradigm, a shrink-wrapped

methodology, ready for passive consumption. It is, rather, a set of perspectives and approaches, linked by a radical vision of use-oriented design" (p. v). This notion of continual experimentation and exploration with scenarios provides motivation for this research.

Additional motivation for using scenarios in this research is that scenarios provide an advantage from a cognitive standpoint with respect to UCD. Scenarios can be designed to target users' needs by letting users' directly fuel requirements generation, as opposed to researchers inferring users' intentions when using methods like observation or ethnography. Scenarios used in this manner address the core of users' mental models because interaction information is obtained directly from users. Consequently, users' future interactions with the developed interface will be more effective and efficient because the interface will be in line with representative users' mental models.

1.6 Purpose Statement

The purpose of this research is twofold: to explore how scenarios can be used to support creative thinking about how new technologies can enhance and extend current work practices, and to facilitate creation of new scenario-based workflows to drive both requirements and subsequent evaluations.

This research will contribute to the preexisting body of knowledge on scenarios by testing prior boundaries and preexisting workflows, therefore facilitating a greater understanding of how they can be effectively utilized in a creative manner.

CHAPTER 2. LITERATURE REVIEW

2.1 Situation Awareness

Due to the variety of domains that explore SA, and the mechanisms of investigating SA that have evolved over time, there have been many proposed definitions of this concept. A widely accepted definition of SA, put forth by Endsley (1988), states that SA is "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future." Each aspect of this definition translates into one of three distinct levels. In Level 1, the operator perceives elements in the environment, which can occur through a variety of means depending on the domain and job. Common mechanisms of perception include visual, auditory, tactile, and olfactory senses, or a combination. In Level 2, the operator comprehends the current situation in light of pertinent tasks and goals, based on the integration of elements perceived in Level 1. In Level 3, the operator projects the future status based on comprehension of the situation in Level 2. A well-designed user interface is integral to perception in Level 1 and the progression through subsequent levels to achieve good SA. It is important to note, however, that good SA will increase the probability of good performance, but does not guarantee it (Endsley, 2003).

SA is often discussed within the realm of dynamic systems (Durso & Sethumadhavan, 2008; Endsley, 1995; Gaba, Howard, & Small, 1995; Gugerty, 1997; Stanton et al., 2006) because of the increased challenge in understanding numerous pieces of data and projecting future system states when a system is constantly changing. While important in dynamic systems, SA is also relevant to static systems in that an operator must have an integrated sense of what they perceive in light of their goals (Endsley, 1995). Due to the static nature of the system in this research, the remaining focus will be on principles of SA that are relevant to static systems.

Several aspects of human information processing can impede the process of building and maintaining good SA. The quantity, organization, and presentation of data influence how quickly the user can process information. When data are disorderly or presented poorly it slows down the flow of information to the brain and subsequently reduces the likelihood of gaining good SA. Additionally, working memory and short-

term memory, which can be thought of as where information is manipulated and stored in the short-term, are limited. This space limitation along with the natural decay of information over time can lead to SA failures. The development of mental models (Gentner & Stevens, 1983), or the users' mental representation of the behavior of a system, can help users to store information more efficiently and therefore help improve SA. Unfortunately, the converse is also true – an incomplete or incorrect mental model can lead to misinterpretation and poor SA (Endsley, 2003).

The design of information on an interface significantly affects SA by determining how quickly and accurately the user can interpret it. Endsley (2003) proposed several principles for system design that can enhance the users' SA. Information should be organized around the users' main goals, as opposed to in a technology-oriented manner. Grouping information together that is needed for a particular goal helps to facilitate comprehension in Level 2 SA. Additional measures should be taken to support comprehension by presenting information in such a manner that it requires the least amount of working memory possible (e.g. directly displaying the deviation between a current value and its expected value rather than requiring the user to calculate it). A design should also support global SA, in that the user should be able to easily comprehend at a high level an overview of the situation across all goals. Since mental models can help to improve SA, as discussed earlier, critical cues for activation of mental models should be uncovered and made salient in the design. Lastly, use information filtering carefully, as too much can degrade global SA. Providing controls for the user to filter information is the most advantageous in supporting individual differences.

2.2 Usability

Usability is essential to the field of human-computer interaction (HCI) in developing interactive software that is useful and easy to use. Though the concept of usability may sometimes seem vague, it can be classified by the following characteristics: effectiveness, efficiency, ease of use, learnability, retainability, and user satisfaction (ISO, 1998). Researchers and practitioners in HCI share the goal of achieving high usability for users of computer-based systems. It is important to note that achieving high usability can mean different things to users from different work roles or with varying

levels of expertise. Hartson (1998) describes what constitutes *high usability* to users of varying skill levels by stating that "a good interface design leads the novice user through task performance, follows intermediate user actions with informative feedback, and gets out of the way of expert users" (p. 103).

2.2.1 Usability engineering

Usability engineering is the process of defining, measuring, and thus improving the usability of a system. Wixon and Wilson (1997) identified three conclusions about usability as it stood in the early 1980s. First, a traditional experimental approach was inadequate for designing user interfaces, such as using conventional statistical tests to answer questions such as "what is the best arrangement of cursor (arrow) keys on the keyboard?" Second, to be treated seriously, usability efforts had to adopt the assumptions and language of engineering. This refers to the fitting of usability into a guiding principle of engineering that "the art of the possible within constrained resources" will be executed, as opposed to goal of science to uncover truth regardless of time or cost. Lastly, for usability to be taken seriously, it had to be treated as part of engineering quality, bringing it to the level of other engineering qualities such as performance and reliability. These conclusions show how usability engineering grew out of the need to move usability from the realm of personal opinion to a quantifiable attribute (Wixon & Wilson, 1997).

Usability engineering is a robust approach for incorporating the representative users' viewpoint into system development. It is a mature and established technique that should be considered in any software-based application or web design effort. Usability engineering provides a platform for contributions from both the user and other stakeholders in the design and development process. In general, usability engineering provides a method for getting users and stakeholders involved early in the design of a system, adds value to the overall design process while allowing contributions of stakeholders, and measures the effectiveness of the design effort, therefore providing a chance to reflect and improve (Wixon & Wilson, 1997).

Every powerful method has its pitfalls, and usability engineering has some of its own. One main pitfall is that there is a danger of setting usability goals which do not

match users' concerns. Additionally, because usability engineering can be an elaborate method, it has the potential to be time consuming (Wixon & Wilson, 1997). Targeting appropriate representative users and maintaining flexibility of the methods are ways to avoid pitfalls in the process of usability engineering.

2.2.2 Usability engineering lifecycle model

A guiding process is needed when incorporating usability engineering into software development efforts as a means of dealing with the complex details, temporal relationships, and dependencies seen in software development projects. A pre-defined process provides a repeatable formula for creating a quality product, and reduces risk by externalizing the state of development among the project roles (Hartson & Pyla, 2012). Most researchers and practitioners agree that an interactive development process should be iterative, and therefore should be represented as a "lifecycle" rather than with a defined beginning and end (e.g., classic waterfall process).

This work uses the iterative usability engineering lifecycle model presented by Hartson and Pyla (2012), which is applicable to nearly any area of user interface design. This work employs all four core usability engineering activities: Analyze, Design, Implement, and Evaluate. The model supports some degree of flexibility in that iteration is possible within each activity, in addition to movement backwards to a previous activity.

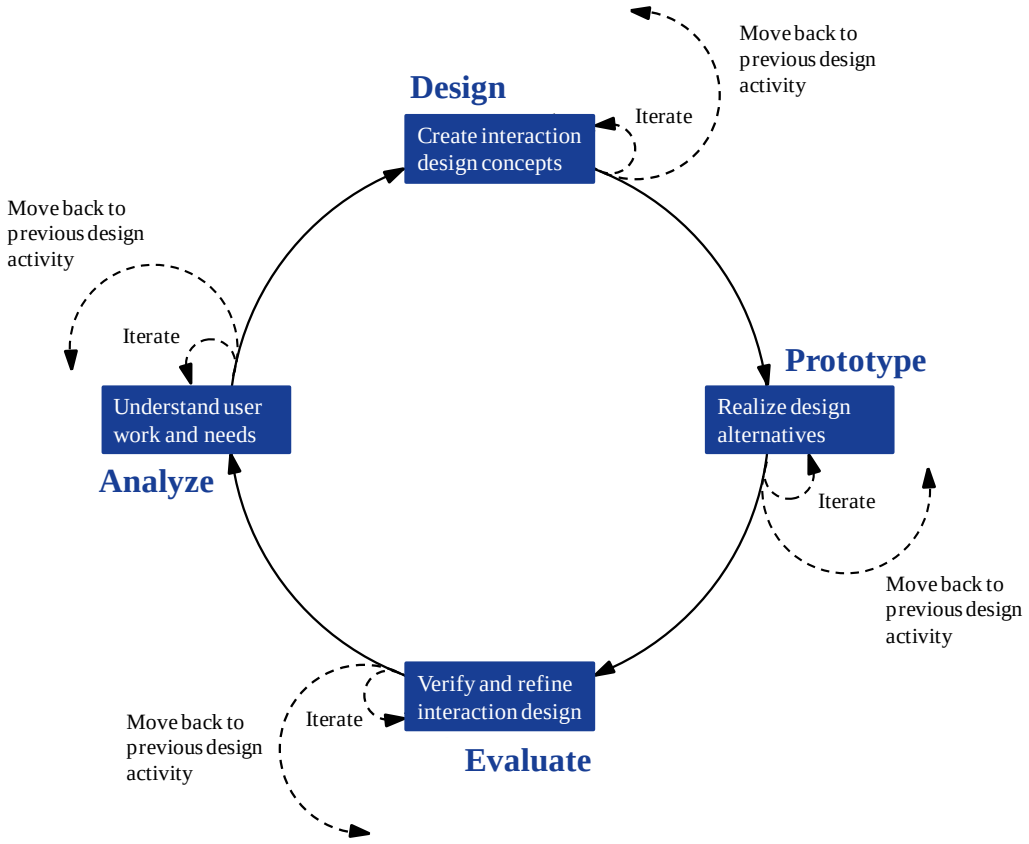


Figure 3. Usability Engineering Lifecycle Model (Hartson & Pyla, 2012), containing four major activities; Dashed lines indicate flexible iteration within and between activities; Used with permission of Elsevier, 2014

Analyze: The analysis portion of the model may include sub-activities such as contextual inquiry and contextual analysis. These sub-activities involve methods such as observation or interviewing of representative users with the aim of understanding user work practices, and identifying underlying themes about work domains. Requirements may be extracted from contextual data which specify what is needed to fulfill user goals. Requirements may include specifications about functionality, as well as the look, feel and behavior of the system. These requirements are used like a checklist to ensure they are addressed during design. The ultimate goal of the Analyze activity is to understand user work and user needs (Hartson & Pyla, 2012).

Design: The design portion of the model encompasses sub-activities which work to create potential design concepts. This may include ideation and sketching, where usability engineers brainstorm and sketch new design ideas. This is a time to explore many different design candidates freely, and not to be constrained by preconceived

notions or previous designs of the system in the case of a redesign (Hartson & Pyla, 2012).

Prototype: The design activity often occurs in parallel with the next activity in the lifecycle model, prototyping. This is a natural partnership since as the designs evolve in the mind and through rough sketches it is helpful to create prototypes as more concrete design representations. Prototypes can be made at a range of fidelity levels, for example paper prototypes at the low fidelity end, and programmed functional prototypes at the high fidelity end (Hartson & Pyla, 2012).

Evaluate: This activity includes verifying and refining the interaction design through a variety of methods, from rapid methods to more rigorous methods. Rapid evaluation methods may include techniques such as design walkthroughs, heuristic evaluations, expert reviews, and formative evaluations. Rigorous empirical methods may involve timing task performance, collecting critical incident data, or having the participant think aloud while performing tasks with the system. The evaluation activity is where one can see if the system meets usability targets. Below is a more detailed discussion about evaluation techniques and methods used in this work.

2.2.3 Usability evaluation

There are various types and methods of usability evaluation, a select few of which are discussed below.

2.2.3.1 Formative evaluation

Formative evaluations help to form the design, and are primarily diagnostic. Their purpose is to collect qualitative and sometimes quantitative data to identify and fix usability problems. Benchmark tasks are used as a means to objectively compare performance and behavior between participants. Benchmark tasks are performed by participants during an evaluation, and encompass a representative spectrum of tasks that the user would perform when interacting with a system in the real world.

Formative evaluations are presented in contrast with summative evaluations, which are used purely to evaluate how well the system meets usability goals. Unfortunately many systems only undergo summative evaluations, at which point it is

very difficult, or too late, to make changes to the system. The need for formative evaluation in interface development has been widely recognized due to the complex nature of the domain (Nickerson & Landauer, 1997).

2.2.3.2 Think-aloud technique

The think-aloud technique is a qualitative data collection technique in which participants express their thoughts about interaction experiences verbally while interacting with the system. Using this method helps the usability engineer to understand the participants' thoughts pertaining to the task, the design, their expectations, strategies, and biases. There is a limit to the amount of information that can be gleaned when simply observing participants' interactions with the system, as a great deal of information about usability issues is hidden in the participants' mind. This technique can elicit information about participants' intentions, motivations, and emotions. The ability to uncover this concealed data is why the think-aloud technique is so valuable (Hartson & Pyla, 2012).

Not only does the think-aloud technique uncover the instinctual thoughts of participants, but it is easy to use for both the usability engineer and the participant. This technique can be used in rigorous and rapid empirical methods, since the only requirement is the inclusion of participants (Hartson & Pyla, 2012).

2.2.3.3 Expert review

An expert review is an evaluation method where interface usability specialists review the system and document the usability issues. This is an easy and fast method that can be used at almost any stage in development, though it most effective early in the process. The goal of an expert review is to explore the design from the user's view, but to see it with an expert's eye. The evaluator is both a participant surrogate and an observer, and must ask themselves questions about what would cause the user problems (Hartson & Pyla, 2012).

2.2.3.4 Cost-importance analysis

While fixing all of the encountered usability issues of a system would be ideal, this is not usually realistic. To prioritize which issues should be addressed, the usability

engineer must look at the cost-effectiveness of each issue. This prioritization is called a cost-importance analysis since it is based on calculating trade-offs between the cost to fix a problem and the importance of getting it fixed (Hartson & Pyla, 2012). While there are a variety of ways to conduct a cost-importance analysis, the core of the analysis involves listing the usability issues in a table, and rating the importance and the cost of each one on quantitative scales. Once each of the usability issues is rated, it is easy to divide the importance rating by the cost rating to determine its priority. The issues are then sorted such that the highest priority is at the top of the table and the lowest priority is at the bottom.

Selecting which issues will be fixed is not quite as easy as working from the top to the bottom of the table. As illustrated in Figure 4, one can see the division of issues into four categories. Category A represents high importance, low cost problems, and category B represents low importance, high cost problems. It is easy to determine that category A issues should be fixed, and category B issues should not. The issues lying in the middle levels of priority are where decisions can become fuzzy. Category C represents issues of low importance and low cost, which depending on the situation may be worth fixing. Category D represents high importance but high cost issues, which is where challenging decisions must be made about just how much time and energy should be devoted to these issues (Hartson & Pyla, 2012).

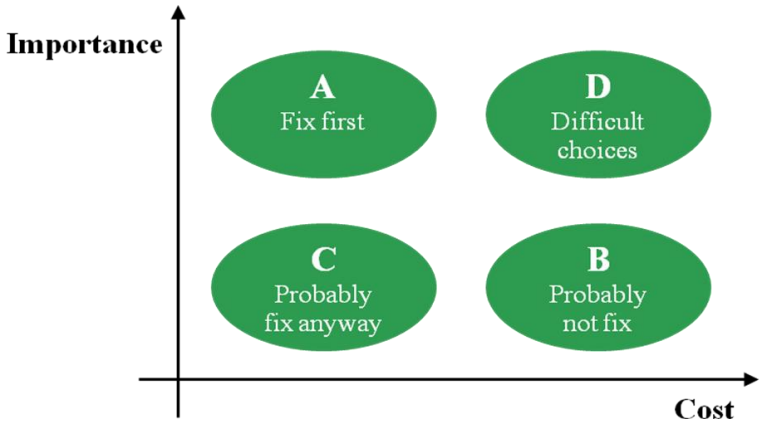


Figure 4. The Relationship of Importance and Cost in Prioritizing Which Problems to Fix First (Hartson & Pyla, 2012); Used with permission of Elsevier, 2014

2.3 Scenario-based Requirements Elicitation

2.3.1 Common applications of scenarios

To understand the concept of a scenario it is helpful to understand how they are commonly applied. Scenarios are used for requirements elicitation early in system design in the fields of software engineering, HCI, and organizational process planning (Antón & Potts, 1998).

Software engineers use scenarios in requirements engineering to help understand the relationship between the system and the environment. Scenarios may also be applied to detailed systems in a more complex manner by depicting interactions between internal system components (Antón & Potts, 1998). In software engineering the collection of scenarios often encompasses the entire functionality of the system. The scenarios may be generated through consultation with representative users, seen in the methodology presented by Hsia et al. (1994), where each possible interaction for each user group is transcribed with the user and organized into a *scenario tree*. Another option is seen in Sutcliffe et al. (1998), where scenario generation is supported by a tool which generates all scenarios based on given use case. From this collection of scenarios, the tool aids in uncovering potential problems within the system. The methods used by software engineers presented here exemplify the use of scenarios to help understand the internal components of a system, as well as their interaction with the environment.

HCI practitioners use scenarios in requirements elicitation to shed light on how the system will support concrete work practices (Antón & Potts, 1998). This is typically a collaborative process between the usability engineer and the representative user, but the stage and the degree to which the user is involved varies. Laporti, Borges, and Braganholo (2009) presented and validated an approach to requirements elicitation based on group storytelling about current and past systems that support a given activity. The representative users went through the predefined approach of telling stories, and converting the stories to scenarios. Each user had a unique role in the conversion process, but everyone participated in storytelling and supported the other people in their duties. Use cases and requirements were elicited from the scenarios primarily by the technical team, but with some support from the stakeholders. Liu, Zhang, and Chen (2012) also

presented and validated an approach to requirements elicitation, but in contrast to Laporti et al. (2009), had the representative users play a more secondary role in scenario creation. The designers first interviewed users to elicit their needs for the system at hand. The users were then asked distinct types of questions aimed at key elements of scenarios. Lastly, information gleaned from users was converted to formal scenario representations, from which requirements were derived. The approaches discussed above demonstrate the use of scenarios to understand how representative users carry out specific work functions with a system.

Scenarios are used within the organizational process planning community to forecast the implications of additions or changes to work practices and information technology support. Scenarios can help make the operational concepts of a system easily comprehensible for stakeholders and analysts, as opposed to more formal techniques sometimes used in organizational process planning. These scenarios aim to be a representation of how groups or individuals using a system would perform a task (Antón & Potts, 1998). Terrell, McNeese, and Jefferson (2004) used scenario and non-scenario techniques to elicit knowledge from employees of a 911 dispatch center. The scenario technique involved presenting the representative users with an emergency scenario, and having them describe the necessary steps to respond to the situation. Results suggest that scenario-based knowledge elicitation produces information tailored to the group's specific culture, which would be an important factor in design in addition to general requirements. This study stresses the importance of the scenario as a tool for facilitating a common language between analysts and stakeholders about their needs.

2.3.2 Diversity and evolution of scenario structure

Scenarios can be created and presented in various forms and levels of detail, and therefore they are best encapsulated in a definition by identifying the range that their common elements can take. Rolland et al. (1998) propose a cohesive framework for defining the vast array of scenarios used in requirements engineering based on four components: form, content, purpose, and life cycle. The form view deals with the level of formality, and the presentation style of a scenario. Scenarios may be more formally presented, such as in mock-ups or prototypes, or simply expressed in natural language.

They may be animated to illustrate the behavior of a system, as opposed to a static presentation such as graphics or texts. Scenarios may also provide a certain level of interactivity with the representative user, meaning the level of influence the user has on the progression of the scenario over time. This broad range of scenario characteristics encompasses the form view of the scenario. The content view is concerned with the type of knowledge conveyed in a scenario, such as events, actions, data, or activities. The content of a scenario varies in level of abstraction, from instance scenarios that define specific agents or events, to abstract scenarios that define broad user groups. System context, referring to the extent to which the scenario includes contextual information from the larger organizations, is also an important aspect of the content view.

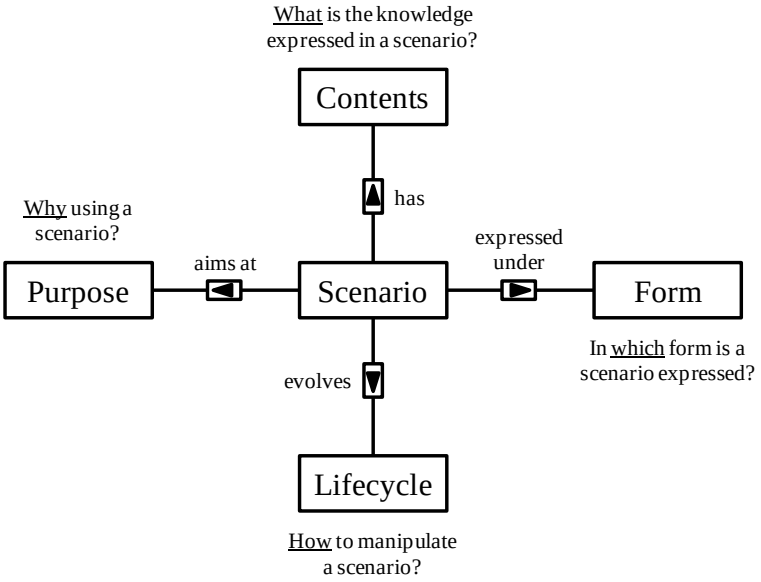


Figure 5. Four Components of Scenarios (Rolland et al., 1998);
Used with permission of Springer, 2014

The purpose view is used to encapsulate the role that the scenario aims to play in the requirements engineering process. Rolland et al. (1998) identify three purposes of a scenario: descriptive, exploratory, and explanatory. Descriptive scenarios walk through a certain process to understand its operations with the intention of capturing requirements. Exploratory scenarios are used in situations when several solutions exist for satisfying a system requirement with the aim of selecting the best solution. Explanatory scenarios are employed when issues are raised that require an explanation about their rationale. Different communities use scenarios for different purposes, as is discussed the previous

section. Lastly, the life cycle view suggests that scenarios should be considered fluid artifacts, capable of evolving in time through the process of requirements engineering. The methods used to guide scenarios through the generation and evolution processes considered in the life cycle view are less commonly found in the literature as compared to defining what comprises a scenario. Due to the inclusion of the life cycle view, and the eloquent encapsulation of the diversity of scenario structure, the classification framework proposed by Rolland et al. (1998) was chosen to be featured as an effort to thoroughly define a scenario.

The beauty of scenarios as fluid artifacts is that they can take on many different forms within the development of one system. Initial scenarios can be extremely rough in that they don't commit to lower level details as to exactly how tasks will be carried out (Rosson & Carroll, 2002). Though the details may be vague, the scenarios provide a working vocabulary between usability engineers and representative users. The evolution of scenarios from informal to formal is thought to be a critical point in the requirements engineering process (Kuutti, 1995). Rosson and Carroll (1995) stress the importance of this evolution in their discussion of the Scenario Browser development tool, which supports an iterative process of scenario development. This iterative process includes generating scenarios, elaborating the scenarios, and continuing to evolve the scenario as new attributes are uncovered. The goal of the Scenario Browser tool, which can be extended to be a worthy goal of the use of scenarios in general, is as follows; a fluid exchange of reasoning between the task level and implementation level of a system should be supported, such that the tasks of representative users can inspire system models, and system models can expand user tasks (Rosson & Carroll, 1995).

2.3.3 Scenario efficacy

There are many advantages of using scenarios, making them an attractive option in requirements elicitation and system design. As discussed in the previous section, scenarios are flexible in their use as they can take on many different forms. Scenarios have been developed as narratives, prototypes, storyboards, acted plays, along with many other forms (Chin, 2004). It is not uncommon to capture scenarios in many forms of media to express different aspects of the scenarios.

Scenarios are accessible to users given that they are typically written in the vocabulary of users. The working vocabulary established between users and analysts provides users a natural and intuitive way to communicating about the system (Chin, 2004). This natural way of understanding and discussing the system simplifies the elicitation process. Not only are scenarios often written in the vocabulary of users, but they are written from the users' viewpoint (Glinz, 2000). This perspective helps users get a better feel for what the system will be like, and is easier to interpret than classic techniques like lists of narrative requirements, or object-oriented class diagrams.

Scenarios are beneficial in reducing the complexity of the system. Contemplating one scenario at a time reduces the number of system aspects the user has to process at once (Weidenhaupt, Pohl, Jarke, & Haumer, 1998). Scenarios afford a decomposition of a system into distinct user-system interactions, where each interactions can be addressed separately (Glinz, 2000). This is a more manageable format for discussing user-system interactions, as users are more familiar with contemplating each action they might take with a system as opposed to the system as a whole.

Scenarios are advantageous in that they ground reasoning in specific details, which is interpreted more easily and thoroughly than abstract ideas. People tend to remember prototypical examples much better than remembering the abstract category it belongs to (Rosson & Carroll, 2002). Due to this focus on reality, small details that may have been overlooked in a more abstract model must be confronted and addressed (Sutcliffe, 2003).

Scenarios enable the user to be involved throughout the system development process, and to envision the system from the users' perspective. Designers may be tempted to use a "solution-first" approach to design, where a design of the system is created before involving potential users. Scenarios provide an excellent alternative to the solution-first approach, as is illustrated in Figure 6.

Hazards of the solution-first approach	How scenario-based design can help
Designers want to select a solution approach quickly, which may lead to premature commitment to their first design ideas	Because they are concrete but rough, scenarios support visible progress, but also relax commitment to the ideas expressed in the scenarios
Designers attempt to quickly simplify the problem space with external constraints, such as the reuse of familiar solutions	Because they emphasize people and their experiences, scenarios direct attention to the use-appropriateness of design ideas
Designers are intent on elaborating their current design proposal, resulting in inadequate analysis of other ideas or alternatives	Because they are evocative and by nature are incomplete, scenarios promote empathy and raise usage questions at many levels

Figure 6. Hazards of Solution-First Approach with Corresponding Scenario-Based Design Solutions (Rosson & Carroll, 2002); Used under fair use, 2014

Rosson and Carroll (Rosson & Carroll, 2002) go even further to suggest that scenarios can be even more effective when users are directly involved in creating them.

Scenarios can also be a powerful mechanism for provoking out-of-the-box thinking while focusing on concrete material. Designers and users alike can become stuck in the familiar, but generating and sharing scenarios which are exceptions to the norm can promote creative thinking (Rosson & Carroll, 2002). Scenarios that are deliberately exceptional can serve as cognitive probes for design (Sutcliffe, 2003). This type of technique may be especially useful in situations where a familiar system is being completely revamped, in order to avoid simply imitating the existing system.

2.3.4 Scenario limitations

Despite many advantages to scenarios, they do have some associated costs. Scenarios even in the simplest form of a narrative can present an overwhelming document-management task when working with a complex system. In other situations, the succinct style of scenarios cannot replace a full-scale task analysis (Carroll, 1995). Scenarios can be very useful as long as usability engineers take the time to determine that they are the appropriate fit for their system.

Scenarios can only be effective if appropriate representative users are involved in development and/or evaluation. Since scenarios are designed to provide information about the context in which the user works, if the wrong users are chosen the scenarios will be misrepresentations of the intended work context. Along this same vein, ensuring

that the users selected for development encompass the range of users of the future system is equally critical (Carroll, 1995). Sometimes these issues arise not because of a misconception of who to select, but simply because there is not enough access or time spent with representative users.

Scenarios can sometimes be too focused in that they do not provide a broad enough view of the system. Concentrating strictly on scenarios can be helpful in clarifying specific issues, but may shroud the interconnectedness between them (Weidenhaupt et al., 1998). When developing scenarios, each representative user will give a personal take on problems they have encountered, making it difficult to elicit commonalities between them. Due to potential discrepancies, specific efforts must be made to ensure consistency, such as following a systematic scenario structuring technique (Glinz, 2000). Sutcliffe (2003) envelops the entirely of this described limitation nicely, noting that becoming consumed with the details in scenarios may result in "not being able to see the wood for the trees."

Representative users may be unreliable narrators about their own experiences, leading to inaccurate scenarios. Users may assume that certain steps in scenarios are known to the usability engineer, and therefore will omit them in descriptions. Additionally, users tend to exaggerate problems or forget abnormal examples. Not only may users be unreliable, but usability engineers themselves can fall into the trap of confirmation bias, by only seeking out examples which agree with their preconceptions. Scenarios should be collected that include exceptions and counterexamples in addition to the norm (Sutcliffe, 2003).

2.3.5 Role of scenarios in requirements documentation

Scenarios are used in a multitude of ways in regards to system requirements. Scenarios may serve as the sole component or as a supplement to a requirements document, or may be used as a stage in an evolutionary process towards a more formal requirements document. In support of the general involvement of scenarios in requirements documentation, Rosson and Carroll (1995) point out that requirements are usually documented as individual features, which leads to voluminous specification documents containing lots of low level detail. This large document of specific operations

does not capture the concept of the whole system as it is experienced by users. Rosson and Carroll (1995) argue in support of scenarios since they help to specify the tasks that the system supports. Since simply including scenarios in a requirements document is conceptually straightforward, the evolutionary process towards requirements documentation will be discussed in greater detail.

Laporti et al. (2009) proposed and executed a method of requirements elicitation beginning with freestyle stories and progressively evolving into more formal representations using scenarios and use cases. This process was done collaboratively, such that process of scenario extraction from stories was lead by usability engineers with the help of representative users. Conversion of the scenarios into use cases was conducted in the same manner, as the usability engineers have the essential knowledge about generation of use cases. Non-functional requirements were then extracted by evaluating the use cases.

Liu et al. (2012) went through a process of requirements elicitation and management which was heavily entwined with the scenarios themselves. Natural language requirements were elicited through interviews with representative users, and became a part of the scenarios. Liu et al. (2012) praise this methodology because when the requirements are interspersed throughout several scenarios it is easy to see the correlations between them.

Potts, Takahashi, and Anton (1994) used an inquiry based approach to develop requirements for a meeting scheduling system in line with the Inquiry Cycle model. The first stage in the Inquiry Cycle is to document requirements. This stage is executed by having stakeholders write down potential requirements: either based on domain knowledge, interviews with representative users, or technical documentation for similar systems. Scenarios were used in acquiring and validating requirements, but are not considered requirements themselves because they only describe the system's behavior in specific situations. Potts et al. (1994) note that despite the narrow focus of scenarios, they are especially useful when challenging tentative requirements in this stage. Scenarios meet the need of the "what-if" questions often asked by stakeholders about the system, in that they provide a specific situation to ground the discussion, while also maintaining awareness about general requirements. The second stage of the Inquiry Cycle is to

discuss the requirements, which is mainly initiated by questions that stakeholders have about a requirement. The answers to these questions result in a clearer understanding of the requirement, and help to uncover ambiguities, inconsistencies, or missing requirements. The final stage of the Inquiry Cycle is requirements evolution, when stakeholders decide to either freeze or change a requirement. A change request is issued to some requirements, and then, based on previous and current discussion, is either refined or untouched. It is important to note that this is not a rigid process model, and shortcuts or modifications should be used as necessary.

2.3.6 Exploration beyond current work practices

Despite the wide array of methodologies incorporating scenarios into requirements elicitation and documentation, there is a lack of a systematic approach, and more broadly an understanding, of the utilization of scenarios within and beyond the realm of current work practices. Carroll (2002) wrote a response to critical reviews of his popular book *Making use: Scenario-based design of human-computer interactions*, entitled "Making use is more than a matter of task analysis", which illustrates this point. While the title is quite revealing as to the nature of his critical reviews and subsequent response, Carroll (2002) details how scenarios used in scenario-based requirements analysis are far more rich than those used in software design. While traditional task analysis seeks to capture a complete description of user tasks, scenario-based design aims to evoke novel insights into current work practices. Carroll does admit that the emphasis on capturing the nuances of human activity using scenarios provides less assurance about internal consistincy. The use of scenarios becomes even more complex when attempting to capture work practices that are the exceptions to the norm, perhaps beginning to edge into the realm beyond common practices due to the emergence of new technologies.

The misinterpretations of Carroll's work show the lack of understanding of how scenarios are used to extend beyond stricly capturing common tasks in a familiar way. This illustrates the need for further investigation into the use of scenarios for requirements elicitation, particularly when incorporating an additional layer of exploration beyond current work practices. As Sutcliffe (2003) notes, biases formed from requirements generated from extreme scenarios are a weakness of scenarios in that they

can misrepresent the viewpoint of a stakeholder. It is important to the communities utilizing requirements elicitation to keep examining and further understanding how to properly manage these requirements.

CHAPTER 3. RESEARCH QUESTIONS

The purpose of this research is to investigate the use of scenarios through various lenses, and the research questions are framed to reflect these different perspectives. Research Question 1 addresses the *levels of scenarios* within the epidemiologist user class. Research Question 2 addresses the *difference in scenario style* between user classes. Research Question 3 addresses the *use of scenarios* throughout the usability engineering lifecycle. Research question are paired with associated hypotheses, which are explored in detail in *Chapter 6. Discussion*.

Research Question 1	What are the differences in the elicited requirements as scenarios move further from extant work practices?
Hypothesis 1	Elicited requirements are increasingly creative as scenarios move further from extant work practices.

Research Question 2	What are the differences in the elicited requirements between structured and free-form scenario groups?
Hypothesis 2	Elicited requirements from structured and free-form scenario groups are comparable with respect to range of MLST functionality and level of creativity.

Research Question 3	Are participant-developed scenarios from the scenario-based interviews effective for use as tasks in formative usability evaluation?
Hypothesis 3	Participant-developed scenarios from the scenario-based interviews are effective for use as tasks in formative usability evaluation.

CHAPTER 4. METHODOLOGY

4.1 Summary of Methodology

To examine these research questions, varying approaches to developing and applying scenarios were explored within the applied context of an iterative usability engineering process for PATRIC's next-generation MLST tool. In short, scenario-based interviews were conducted to collect relevant work-practice information and domain knowledge from representative users (see *Section 4.3.1 Scenario-based interviews*). Requirements distilled from the scenarios and complementary interview questions informed the design of V1 of the MLST interface. V1 then underwent an expert review. Information gleaned from the expert review was used to make improvements to V1, resulting in V2. A formative usability evaluation was conducted on V2, which at the core consisted of representative users performing a set of benchmark tasks using the evolving MLST tool. A cost-importance analysis of usability issues discovered by the formative evaluation was conducted to determine which aspects of V2 should be modified. V2 was iteratively refined and updated accordingly to create V3. A visual summary of this overall approach is presented in Figure 7, which superimposes the specific elements of this approach onto Hartson and Pyla's (2012) usability engineering lifecycle model.

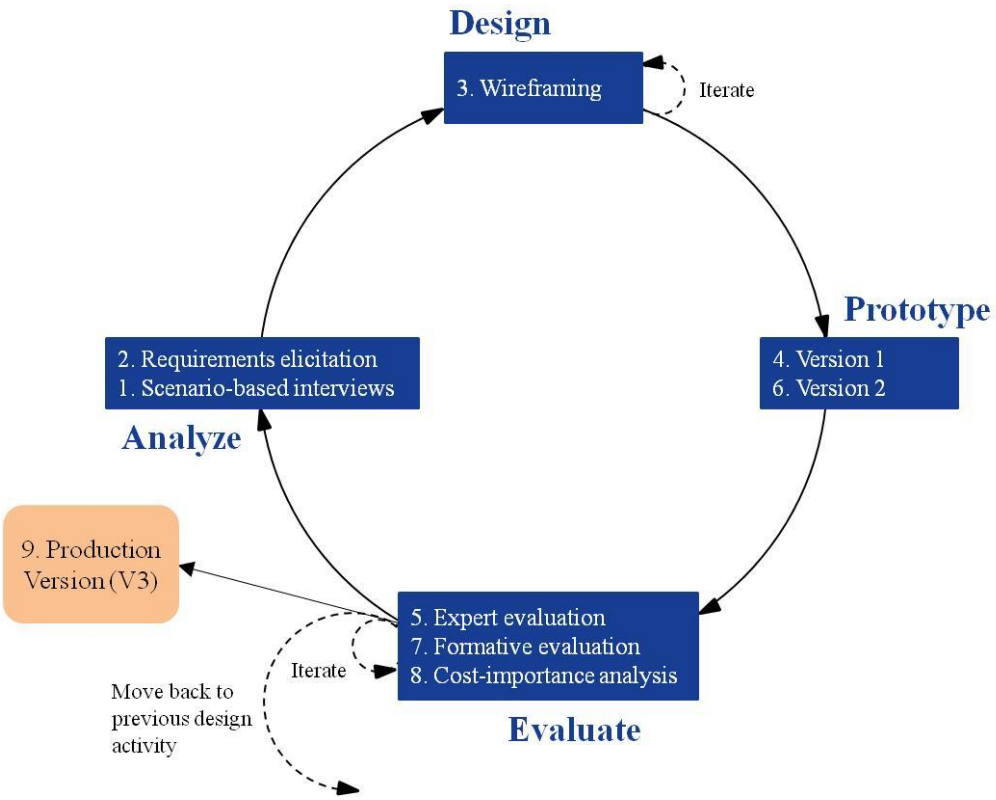


Figure 7. Summary of Methodology

The three hypotheses are investigated in different stages of the methodology. *Hypothesis 1* and *Hypothesis 2* pertain to the requirements elicitation stage, which specifically examine requirements from different levels of scenarios and requirements from different styles of scenarios, respectively. *Hypothesis 3* pertains to the formative evaluation stage, specifically examining the use of participant-developed scenarios as tasks in the formative evaluation.

4.2 Participants

Participants were four working professionals who use MLST to achieve specific work goals in their respective domains. The balance between epidemiologists and molecular biologists was considered due to the distinct manner in which each user class utilizes MLST. Participants 1 and 4 comprised the epidemiologist user class, and participants 2 and 3 comprised the molecular biologist user class. The same four participants were used for both the scenario-based interviews and the formative usability evaluation since participation in the former activity did not bias the latter. This sample

size aligns with empirical studies that examine scenarios which typically employ two to four participants in each user class (Neale & Kies, 1996; Terrell et al., 2004), in addition to studies that suggest that involving four or five participants in the test phase of usability evaluation uncovers 80% of the usability problems, including usability problems of the highest severity (Virzi, 1992).

Table 1. Participant Demographics

Participant	Current Occupation	Training	Time in Field	MLST Experience
1	Infectious Disease Epidemiologist	Epidemiology Microbiology	9 years	MLST was main tool used as graduate student and for PhD dissertation
2	Research Scientist	Microbiology Genomics	5 years	MLST is one of many subtyping methods used
3	Research Microbiologist	Microbiology Microbial population genetics Immunology	23 years	MLST is one of many subtyping methods used
4	Applied Epidemiologist	Epidemiology	5 years	MLST is one of many subtyping methods used

4.3 Procedure

4.3.1 Scenario-based interviews

The scenario-based interviews were conducted over the telephone and lasted approximately 45 minutes. The participants engaged with the research team in an interview targeting their common work practices and domain knowledge, using standard interview style questions as well as scenarios (see *Appendix B: Scenario-based Interview Protocols*). The research team took notes during the interview detailing the information provided by the participants. Notes taken detailing the scenarios elaborated on, or generated by participants were written in full narrative scenarios form after the interview to facilitate subsequent requirements elicitation.

Epidemiologist

Since one intent of scenario use in this study was to facilitate out-of-the box thinking about the addition of original functionality to the system, it was inappropriate to present participants with a complete scenario for evaluation, as is common in many studies. Instead, participants were presented with "seed" scenarios, which are a narrative

description of an issue that they might encounter in their daily work practice. The seed scenarios were aimed at certain use cases, either commonplace or radical, and the intent was for the participant to complete and extend the seed scenarios by describing how they would approach this situation if faced with it in their work. Scenarios were presented to participants in order from most commonplace to most radical.

An expert in the field was used to create three scenarios, using the top down decomposition approach (Weidenhaupt et al., 1998),which represent common situations that the participant would be accustomed to, in addition to situations that the participant is not used to performing given the current functionality of similar systems. Usability engineers aided in selection and refinement of seed scenarios. Due to the relative simplicity of earlier versions of the system, and the fact that the full extent of system functionality is unknown, it was not necessary to generate scenarios that encompass the entirety of the system.

Molecular Biologist

Due to the highly variable ways that bacterial molecular biologists use MLST tools, it was not logical to present them with seed scenarios. In the scenario-based interviews with molecular biologists, the participants narrated scenarios they have encountered in the past. Probe questions (e.g., What challenges did you encounter? What resources did you turn to for help?) were posed to participants during their narration aimed at encouraging them to think further about other ways they could pursue the research they were describing in their scenarios (see *Appendix B: Scenario-based Interview Protocols*).

4.3.2 Requirements elicitation

Requirements were distilled from information gleaned from the scenario-based interviews and from domain knowledge using the inquiry cycle detailed in *Section 2.3.5 Roles of scenarios in requirements documentation*. Requirements elicited from each of the user classes were kept in two separate requirements documents to monitor how the different scenario approaches progressed throughout the usability engineering lifecycle. The resulting sets of requirements were used to drive initial conceptual design and

wireframing. The sets of requirements along with iteratively refined wireframe designs were given to the software developer to develop V1.

4.3.3 Wireframing

Wireframing was the main design activity employed for this research, and iterative sketching in particular was heavily used. Various sessions of sketching on a dry-erase board took place to brainstorm and organize preliminary design ideas. Sketching helped to facilitate ideation and discussion by providing a visual medium as a common group foundation. During sketching there were many instances of drawing a manifestation of a particular idea, and afterward erasing and re-drawing it. These mini-iterations exemplify the flexibility and creativity of sketching, and are essential to the ideation process. The final sketches from each sketching session were photographed, and used as inspiration for more formal wireframe creation. The formal wireframes were created in Microsoft PowerPoint and took concepts and design cues from the sketches, but fleshed out the layout and behavior of the interfaces in greater detail. The formal wireframes also went through a series of iterations, mainly as the interaction between the different interfaces became more apparent. Other, more finely tuned, iterations came later as specific interactions with the interface were examined.

4.3.4 Expert review

An expert review, as described in *Section 2.2.3.3 Expert review*, was conducted on V1. Each of two usability specialists performed an individual expert review of V1. Afterward, they collaborated to reconcile differences between the two documents. Common usability issues between the two documents, or usability issues that the two parties agreed were appropriate, were aggregated into one final expert review.

The uncovered usability issues (e.g., lack of cohesion between all components of interface) and corresponding design recommendations (e.g, title for each component should be same size, color, font) were subsequently delivered to the software developer. The software developer made the appropriate revisions based on the expert review to iteratively refine and improve the MSLT prototype tool, resulting in V2.

4.3.5 Formative usability evaluation

A formative usability evaluation, as described in *Section 2.2.3.1 Formative evaluation*, was conducted by having participants work through benchmark tasks using V2 (see *Appendix C: Formative Usability Evaluation Protocol*). Participant-generated scenarios from the scenario-based interviews were distilled into tasks by identifying the MLST functionality that participants described utilizing, and creating tasks prompts that evoked a similar interaction with V2 of the interface. These tasks were combined with additional tasks that were regarded as appropriate for a more complete analysis of the interface.

Participants were instructed to use the concurrent think-aloud method (Ericsson & Simon, 1998) during task execution to provide insight into their thoughts while interacting with the interface. After completion of each task, follow up questions were asked to debrief about any interesting or unclear verbalized thoughts to achieve a deeper understanding of the interaction. Though this method may lose some deeper qualitative data as compared to questioning the participant during the task, the concurrent think-aloud method was appropriate for this research due to the primary purpose being a usability evaluation.

Qualitative data collection included documentation of critical incidents (Flanagan, 1954), usability insights (Nielsen, 1994), and any additional vocalizations from the participant deemed relevant. *Quantitative* data collection included ratings of ease-of-use and usefulness for each relevant task (see *Appendix C: Formative Usability Evaluation Protocol*), in addition to task time (i.e., total duration the participant spends on the task) and a binary measure of accuracy (i.e., 0 = incomplete/incorrect attempt of task, 1 = successful completion of task) to determine task effectiveness and participant efficiency.

4.3.6 Cost-importance analysis

A cost-importance analysis was conducted on the usability issues uncovered in the usability evaluation. Each usability issue was rated on a scale of one to five for both importance and cost. The priority of each usability issue was determined using the following formula:

Priority = Importance / Cost

All usability issues with a priority of one or higher were given to the software developer to develop V3. Implementing solutions to the usability issues, and the resulting V3 MLST tool generated from these implementations are beyond the scope of this research.

CHAPTER 5. RESULTS

The following subsections include detailed results for each stage of iterative usability engineering process described in *Chapter 4. Methodology*. The findings presented in *Section 5.1 Requirements* ultimately refute *Hypothesis 1* and *Hypothesis 2*. The findings presented in *Section 5.5 Formative Usability Evaluation* support *Hypothesis 3.*

5.1 Requirements

The scenario-based interviews resulted in a total of 25 requirements, with seven requirements overlapping between the user classes. The epidemiologist user class yielded 14 requirements and the Molecular biologist yielded 18 requirements. In addition to overlap in requirements between user classes, there was overlap within user classes, though this is not evident from the requirements documents. The epidemiologist requirements generated from the structured scenario portion of the interviews are marked with an asterisk to track requirements for *Research Question 2*.

Table 2. Elicited MLST Tool Requirements

Epidemiologist Requirements	Molecular Biologist Requirements
Ability to select a specific organism*	Ability to select a specific organism
Input individual loci or concatenated MLST sequence*	Input individual loci or concatenated MLST sequence
Batch sequence query*	Batch sequence query
Identify MLST profile*	Identify MLST profile
Large and complete data sets*	Large and complete data sets
All parts of genome available for analysis*	All parts of genome available for analysis
Ability to investigate virulence factors*	Ability to investigate virulence factors
Include contact information to request more data	Integration of different species into one MLST tool
Substantial metadata	Not a steep learning curve
Side-by-side information visualization for increased situation awareness	Contribute to the MLST database without advanced computer science knowledge

Epidemiologist Requirements	Molecular Biologist Requirements
Message board	Display close relatives
Map depicting where an organism with the MLST profile has been located	Display closely related MLSTs by individual gene loci
Guidance for novices	View strain type in phylogeny
Tie in bioinformatics tools/visualizations into existing public health data	Display percent nucleotide match within each sequence type
	eBURST software functionality
	Input whole genome sequence
	Ability to look for signals that DNA are imported from different species
	Links to publications on related genomes

5.2 Sketches and Wireframes

The resulting sets of requirements were used to drive initial conceptual design (i.e., sketching and creating workflows) and wireframing. The sets of requirements along with iteratively refined wireframe designs were given to the software developer to develop V1.

5.2.1 Sketches

The first sketch, depicted in Figure 8, shows a broad scheme for the multiple ways users can access the core functionality of the tool, including paths that support browsing existing PATRIC data, in addition to methods of bringing one's own data for comparison. The red box outlines a brainstormed list of data, links, information and user actions needed to support the requirements.

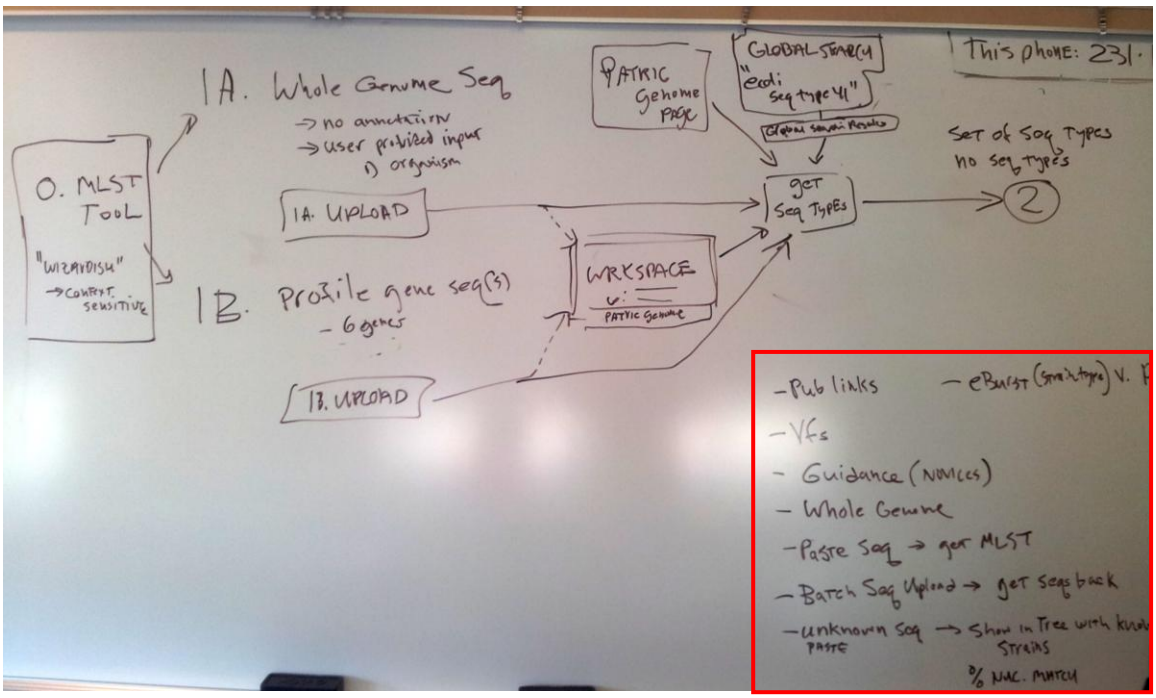


Figure 8. Broad Scheme of Access to Core Functionality

Figure 9 illustrates a general conceptual model for the tool. The top left portion of the sketch lists potential mechanisms to select a specific MLST profile. An MLST profile is comprised of an organism name and a sequence type (e.g., E. coli ST 131). The red box outlines three main areas of functionality used to learn more about the selected MLST profile. The sketch goes into further detail about the first main area of functionality, Profile Genes, by including drawings and a list of potential features.

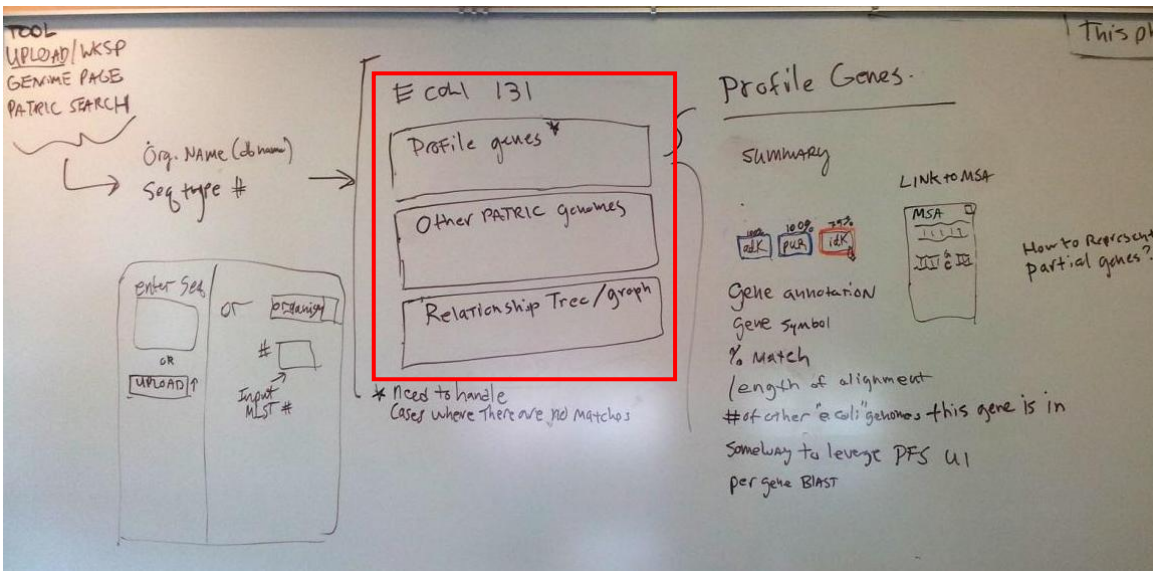


Figure 9. Sketch of General Conceptual Model and Profile Genes Concept Details

Figure 10 and Figure 11 show the second and third areas of functionality, respectively, of the MLST tool. The Other PATRIC Genomes functionality includes a genome table, outlined in red, and a map. The Relationship Tree functionality includes a tree in the top portion of the sketch illustrating the evolutionary history of relevant genomes, paired with relevant metadata (e.g., location, date collected).

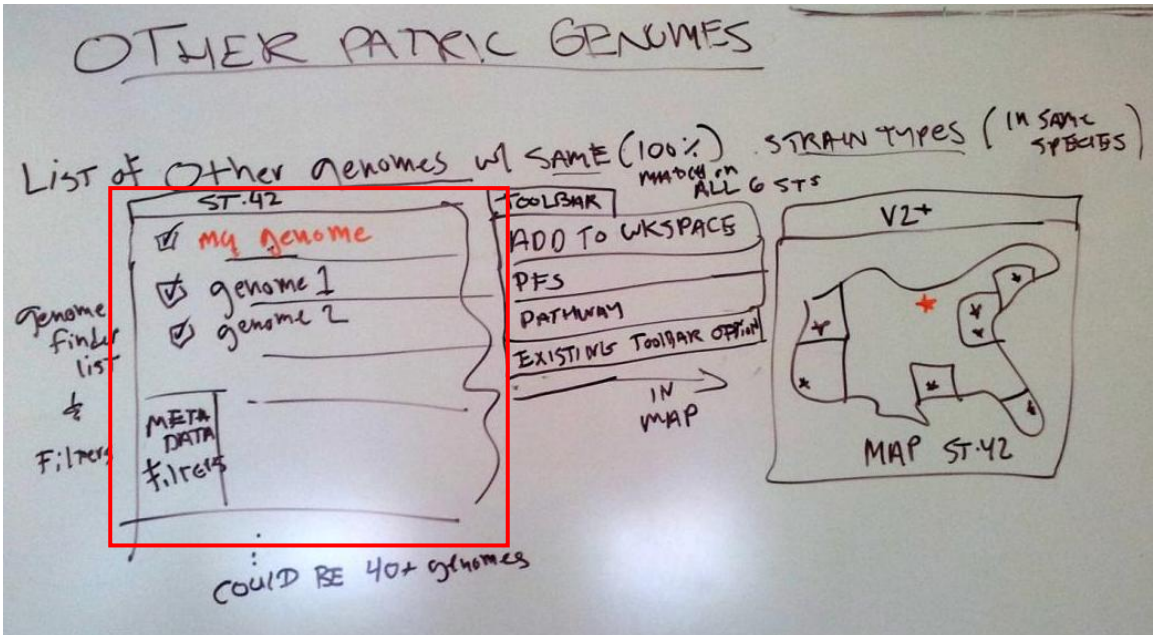


Figure 10. Sketch of Other PATRIC Genomes Concept

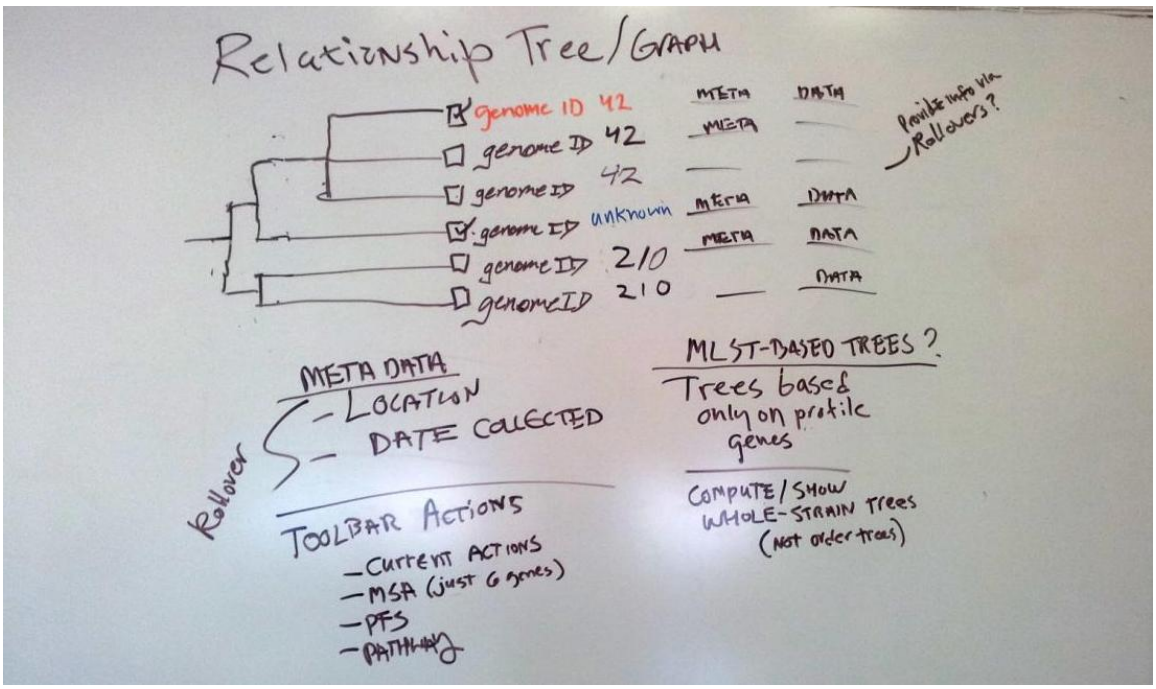


Figure 11. Sketch of Relationship Tree Concept

Lastly, Figure 12 illustrates the unification of the previously sketched concepts into one interface, which would be the landing page for a particular sequence type (ST). Outlined in red is the Profile Genes functionality, with a genome table in the bottom left and the Relationship Tree functionality in the bottom center.

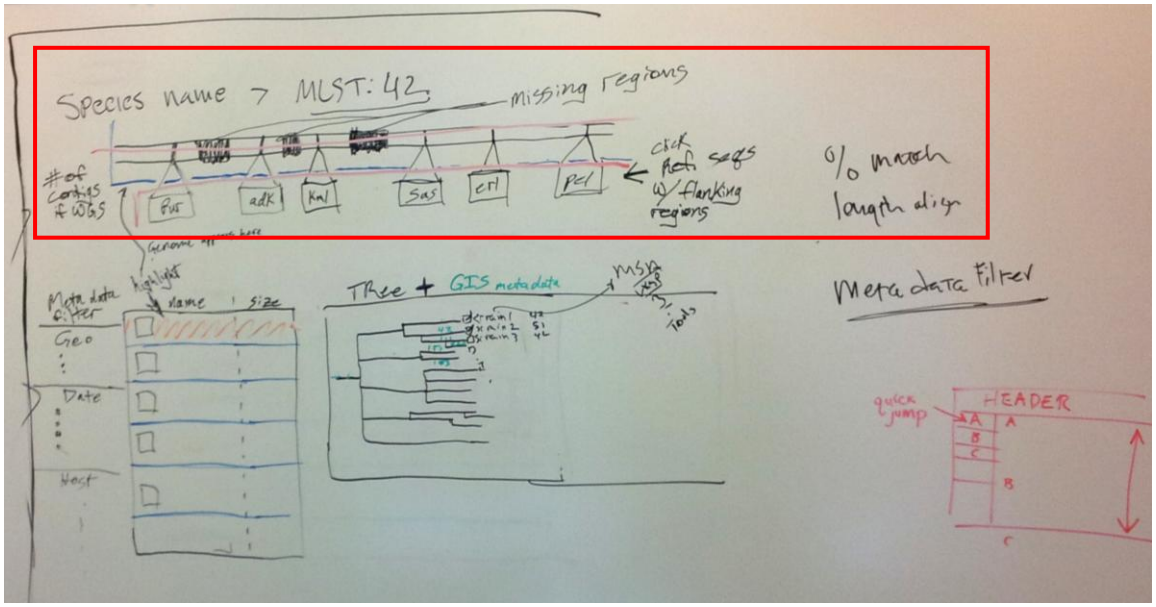


Figure 12. Sketch of Landing Page Design

5.2.2 Workflows

The sketching activity was useful for brainstorming, in a broad sense, potential features and how users might interact with the interface. However, the details of specifically which features would be on each page, and how users would navigate between pages still needed to be fleshed out. During the early stages of wireframing we identified two core workflows that both user classes would need, which are depicted in Figure 13. Workflows were reviewed by four domain experts to further ensure that the workflows would align with users' mental model of interaction with an MLST tool.

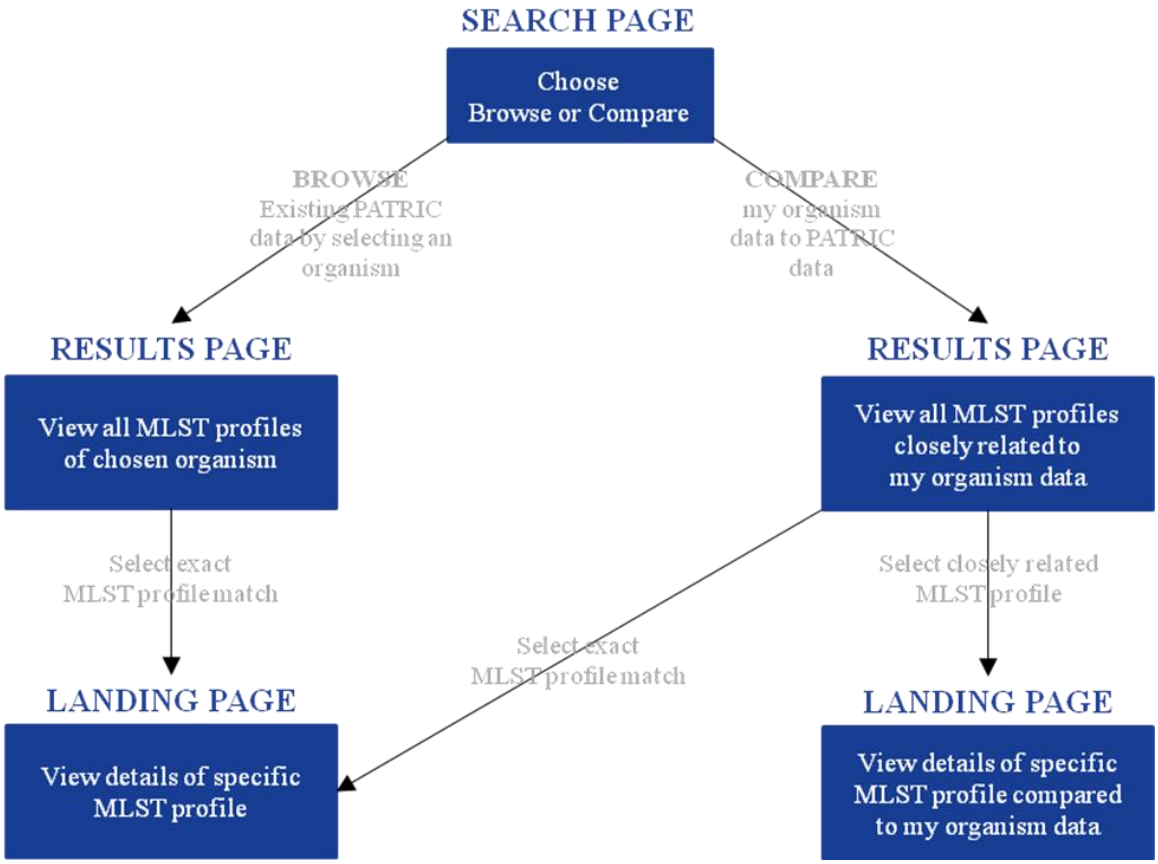


Figure 13. Workflows of MLST Tool

Each blue box represents a different page in the interface. The name of each page is shown at the top of each blue box, and a description of the contents of each page is inside the blue box. A description of the action taken to move to a specific page is in gray on top of the arrows.

Both workflows begin at the Search Page, where users choose to either Browse existing PATRIC data, or Compare their own data to PATRIC data. These two different

utilizations of the tool will be termed the Browse mechanism, and the Compare mechanism, respectively. The next page users reach is the Results page, where they view a table of MLST profiles in response to their search criteria. On the Results page, users select a specific MLST profile that they investigate further on that specific MLST profile's Landing page.

5.2.3 Wireframes

The wireframes are displayed in pairs, showing the Browse and Compare mechanisms side-by-side for each page. Figure 14 depicts the Search page of the MLST tool when used for either browsing existing PATRIC data (i.e., Browse mechanism), or bringing in one's own data for comparison (i.e., Compare mechanism). Once a sufficiently robust and accurate version of the Search page wireframes were created, some fine-tuned iteration were made. These iterations include flipping the radio buttons for selecting My Data and PATRIC Data to place them in a more logical order. The order of the My Data data input boxes, outlined in red, were also reversed due to a higher likelihood of utilization of the Upload mechanism as opposed to the Paste mechanism. Descriptive instructional language was also added to these data input boxes to help guide the user in properly executing a search. Additionally, the Organism list was refined to include only organisms for which there is data, so as not to mislead or confuse the user.

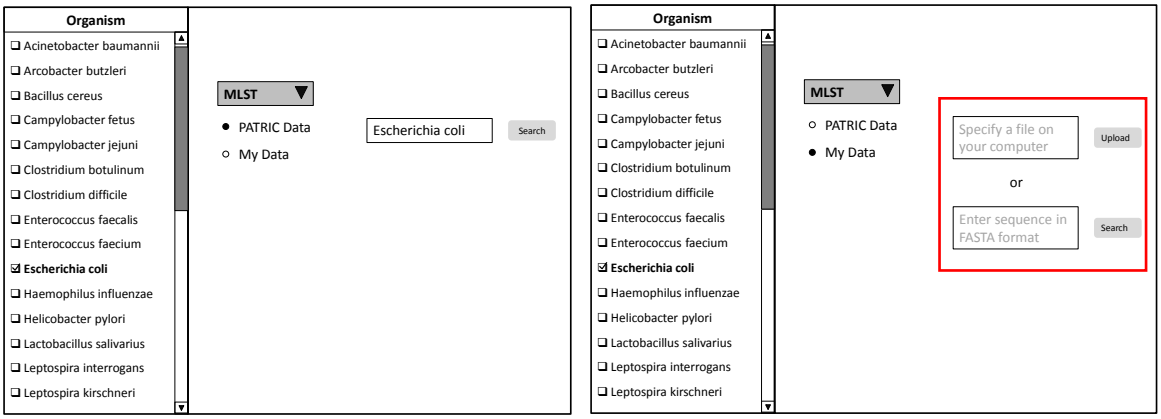


Figure 14. Search Page Wireframes for Browse Mechanism (left) and Compare Mechanism (right)

The next set of wireframes, depicted in Figure 15, display the Results page when using the Browse mechanism and the Compare mechanism. While the table itself is informative, additional contextual information, outlined in red, was added to help orient the user.

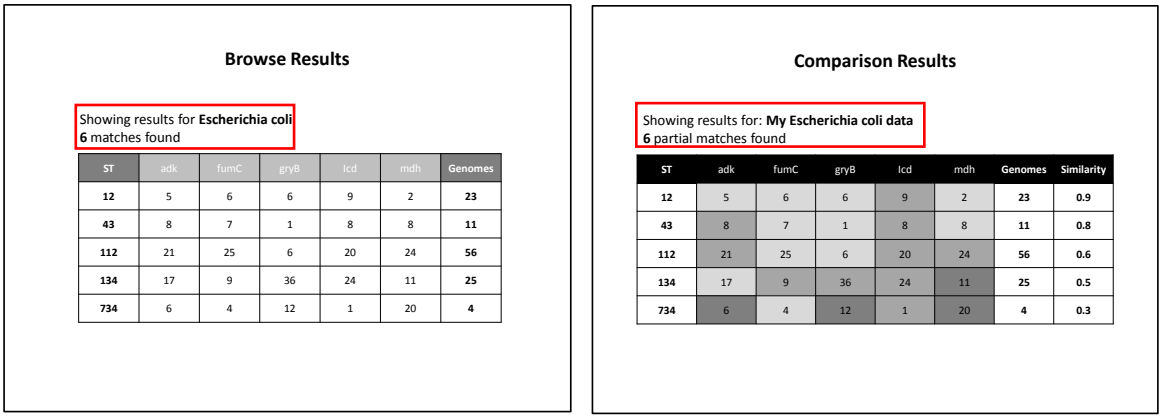


Figure 15. Results Page Wireframes for Browse Mechanism (left) and Compare Mechanism (right)

Figure 16 shows the Landing page for a specific MLST profile (e.g., E. coli ST 131) for the Browse and Compare mechanisms. The Landing pages required the most iteration due to their considerable complexity, specifically the details of the allele options visible in the upper left panel of the Compare Landing page. The components of the allele options were each designed individually, but were organized together to appear visually consistent and simplify the interface. This organization included adding additional functionality to some aspects of the allele options, and updating the language for further clarity. The upper right panel of the Browse Landing page shows a Phylogeny (another word for a Relationship tree as described in *Section 5.2.1 Sketches*) which shows the evolutionary history of relevant genomes. The upper right panel of the Compare Landing page shows an eBURST diagram, which is another way to visualize relationships between genomes. The bottom portion of both Landing pages shows a genome table.

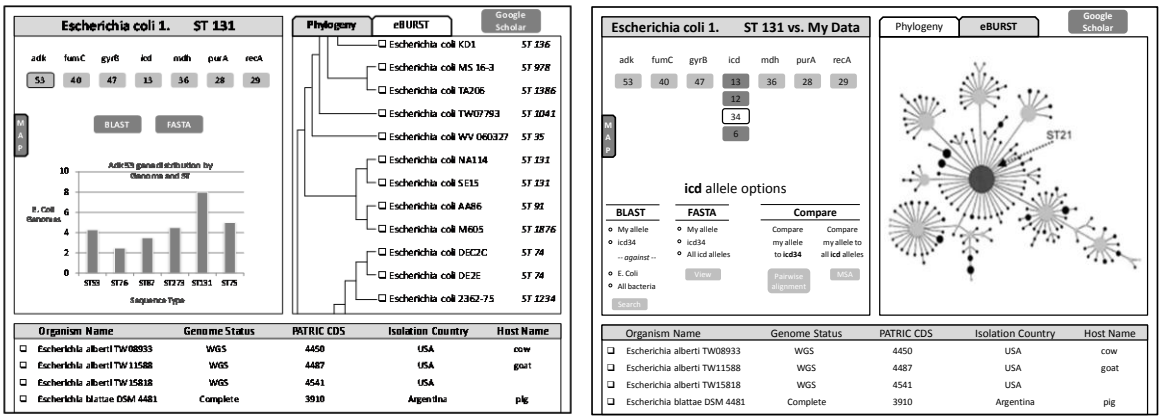


Figure 16. Landing Page Wireframes for Browse Mechanism (left) and Compare Mechanism (right)

5.3 Expert Review

V1, created by the software developer using the elicited requirements and iteratively refined wireframes, underwent an expert review by two usability specialists. Though V1 was functional, the expert review was instrumental in improving usability. The expert review provided the opportunity to view the interface through the eyes of the user, but with the eye of an expert. For the sake of context and continuity, the complete expert review appears below.

Table 3. Expert Review Document

SEARCH PAGE	
Usability Issue	**Recommendation**
Language describing the two options of data source may be confusing to users.	Revise language to make the selection of data source clearer. For example: **Browse** PATRIC MLST data **Compare** Your sequences to PATRIC MLST data
RESULTS PAGE	
Usability Issue	**Recommendation**
The alignment scores visible when hovering over an allele are not consistent.	Put all alignment scores in the format "x.xx" as opposed to "x.."
The legend for the similarity color-coding is visible on the results page when there is no color in the table.	Only display the legend when the table has colored values so as not to confuse the user.
LANDING PAGE	

Usability Issue	Recommendation
General	
Lack of cohesion between all components of interface.	Use the same background coloring for each component (e.g. change phylogeny to white background). Title for each component (profile genes, phylogeny/eBURST, table) should be same size, color, and font (16 point font, black, bold).
Components stay the same size when the interface window shrinks, making it difficult to work with.	Redraw interface when window is resized to fill the browser window.
Profile genes	
Introductory text pop-up should not include the word "please", and should refer to the colored nature of the allele boxes instead of the numbers due to the fact that the color is the novel feature.	Introductory text pop-up should read "To start, select an allele by clicking a colored box above."
It is not clear which allele is selected after it is initially chosen.	Selected allele should maintain outline box when selected.
Mouseover of allele doesn't show full word "alignment", only "alignm."	Ensure that all mouseovers render properly.
The labels on the x-axis of the allele distribution bar graph do not properly convey what the axis represents.	Change a plain allele number, for example "45", to "ST45."
The allele options title does not stand out enough, which may cause users to be unsure of how to interact with and interpret the allele options.	Consider making the text larger and/or distinguishing it more clearly in another way (i.e. faint horizontal lines on either side of the title).
BLAST/FASTA/Compare headings are not distinguished enough.	Highlight the headings in a light, but noticeable color.
Language distinguishing the unknown allele in BLAST/FASTA/Compare is too informal.	Instead of "My **adk** allele", use "Unknown **adk** allele."
Profile gene panel is not consistent when using the "Browse" and "Compare" functionality.	Provide the same options and UI for both the "Browse" and "Compare" scenarios to reduce confusion for users.
Phylogeny/eBURST	
It is difficult to determine if the Phylogeny or eBURST tab is selected.	Consider coloring the selected tab the same color scheme as the section titles in order to bring attention to it, and color the unselected tab a very light blue. Additionally, the selected tab font should be bold, and the unselected tab should be in regular font.
Color key buttons have the same look as component titles, which may cause confusion to the user.	Distinguish the color key button in a different manner.
It is hard to get a full sense of what the visualizations are.	Incorporate the ability to zoom in and out of the phylogeny and eBURST.
It may be difficult for users to locate a specific genome.	Consider adding "find" functionality.
The checkboxes next to each genome in the phylogeny currently serve no purpose.	Create functional checkboxes which enable the user to select specific genomes.

The phylogeny should incorporate more interactive capabilities for optimal use.	Incorporate the ability to use branches to select multiple genomes, and the ability to collapse and expand branches.
Genome table	
The user cannot get a full sense of the extent of the table results from the heading alone, which will be the only indication "above the fold."	Include the number of records in the title.
Column management is hidden and redundant.	Add a more obvious mechanism for showing/hiding columns, for example a dedicated button on a table toolbar.
The bottom right corner of the table displays the text "No data to display", which is inaccurate when data is available.	Remove this text.
Some mouseovers on the table controls do not display fully.	Ensure that all mouseovers render properly.
Miscellaneous	
Map handle is too large and does not afford an obvious action.	Handle should be smaller or should be relocated, and be shaped in such a way that it affords a more obvious action by the user (i.e. a sideways tab).
Map handle has the same look as component titles, which may cause confusion to the user.	Distinguish the map handle in a different manner.
The Google Scholar button has the same look as component titles, making it blend in with the rest of the interface. Additionally, it is unclear what exactly will happen when clicking on the button.	Instead of a button, have a Google Scholar link with an attention grabbing icon and a more explicit title.

5.4 V2 Screen Shots

The expert review was given to the software developer, who used the document to refine and improve V1, resulting in V2. The following images are screen shots of V2 of the MLST tool. The screen shots are broken down into the Browse mechanism and Compare mechanism workflows.

5.4.1 Browse mechanism

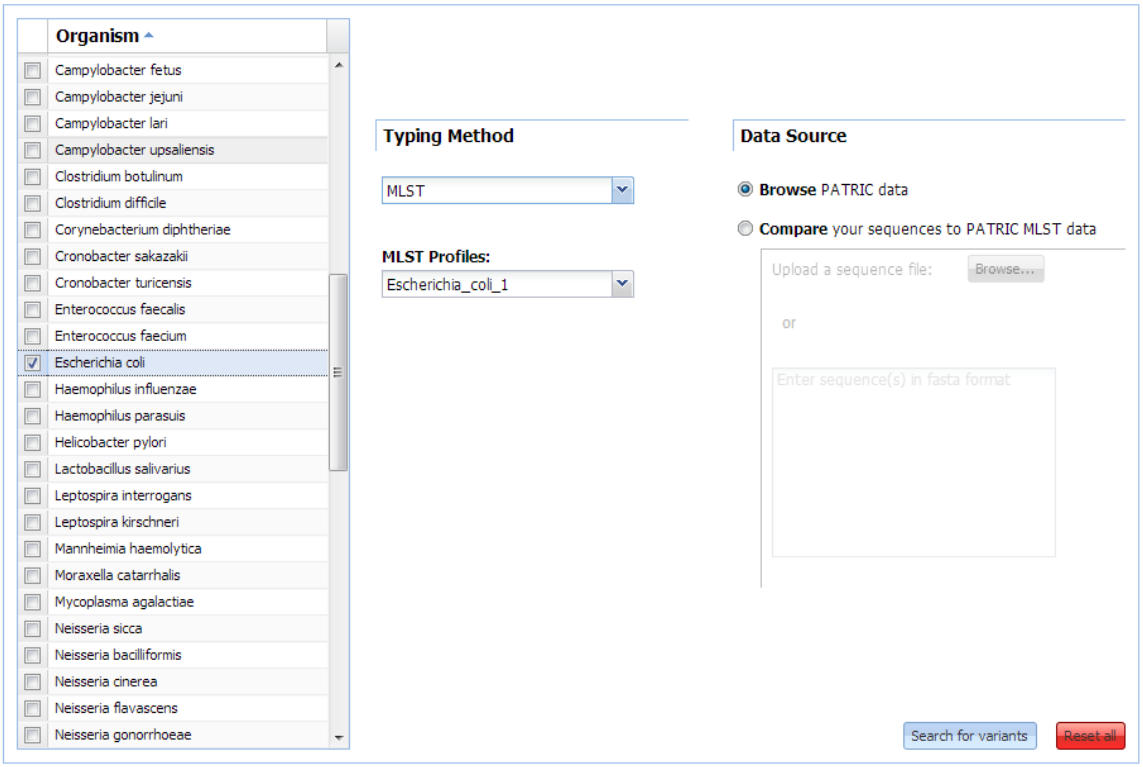
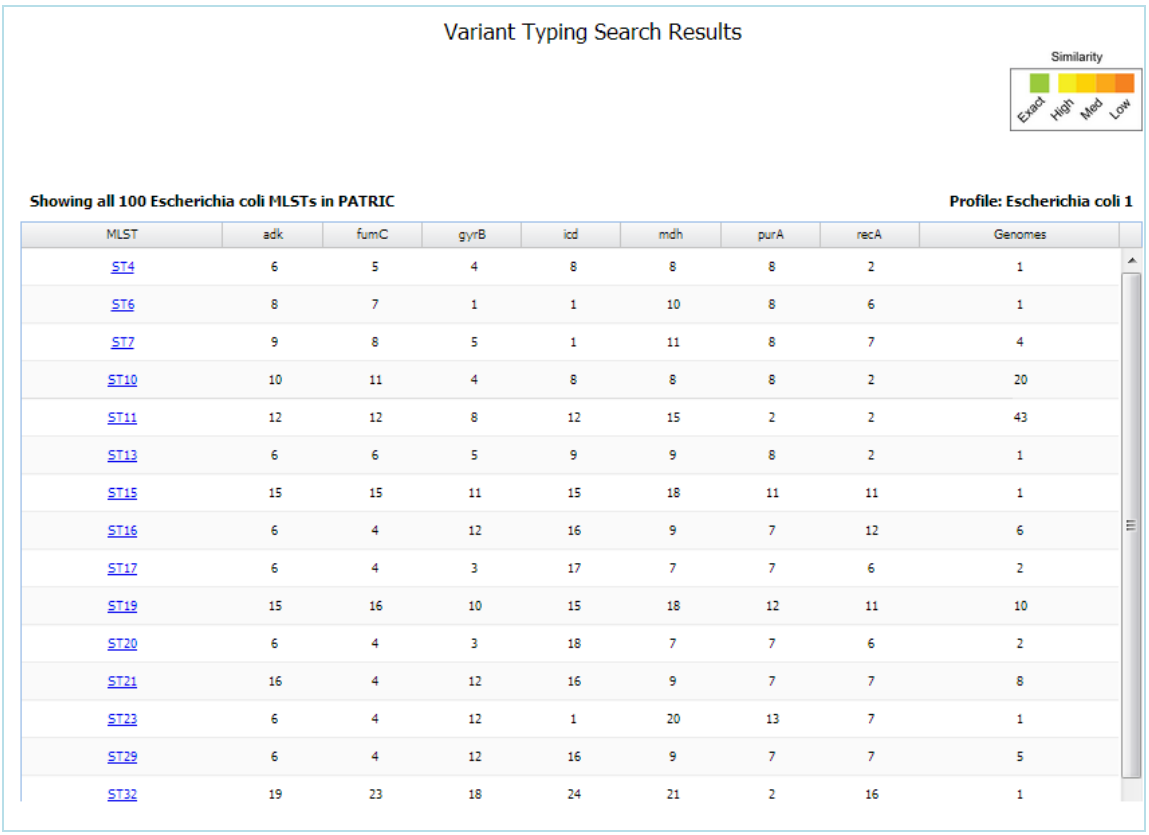
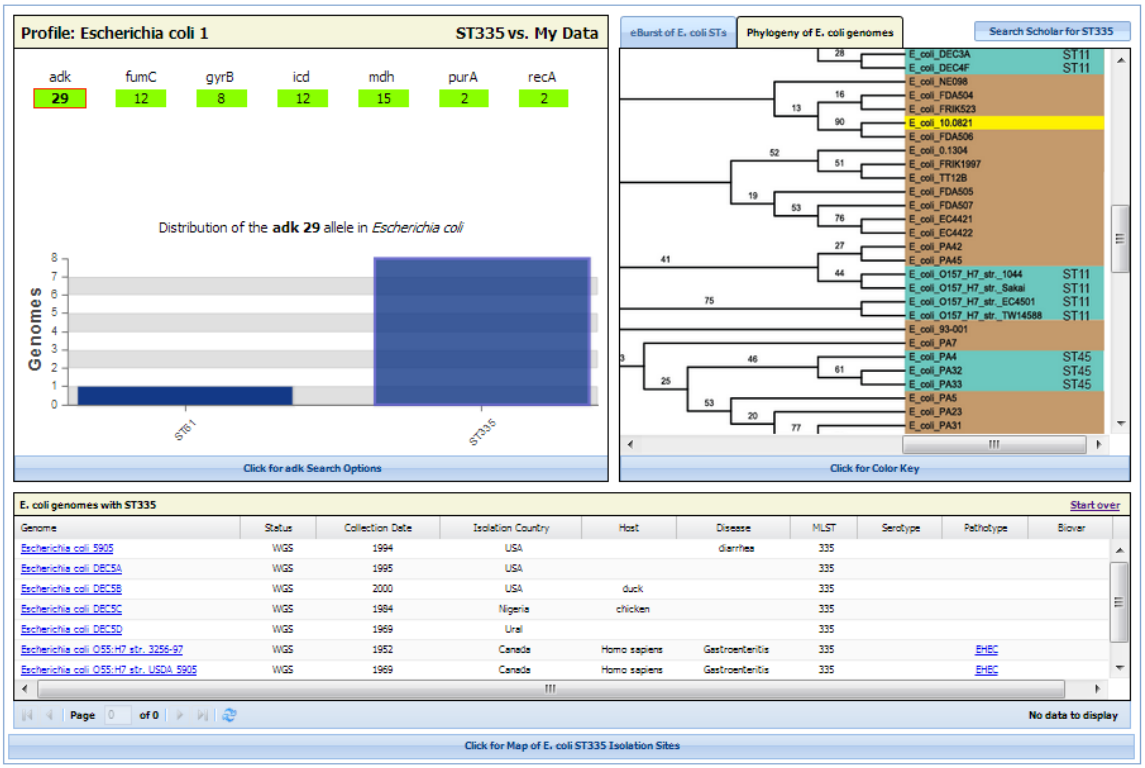


Figure 17. Screen Shot of Search Page for Browse Mechanism



MLST	adk	fumC	gyrB	icd	mdh	purA	recA	Genomes
ST4	6	5	4	8	8	8	2	1
ST6	8	7	1	1	10	8	6	1
ST7	9	8	5	1	11	8	7	4
ST10	10	11	4	8	8	8	2	20
ST11	12	12	8	12	15	2	2	43
ST13	6	6	5	9	9	8	2	1
ST15	15	15	11	15	18	11	11	1
ST16	6	4	12	16	9	7	12	6
ST17	6	4	3	17	7	7	6	2
ST19	15	16	10	15	18	12	11	10
ST20	6	4	3	18	7	7	6	2
ST21	16	4	12	16	9	7	7	8
ST23	6	4	12	1	20	13	7	1
ST29	6	4	12	16	9	7	7	5
ST32	19	23	18	24	21	2	16	1

Figure 18. Screen Shot of Results Page for Browse Mechanism



Genome	Status	Collection Date	Isolation Country	Host	Disease	MLST	Serotype	Pathotype	Biovar
Escherichia coli 5905	WGS	1994	USA		diarrhea	335			
Escherichia coli DEC5A	WGS	1995	USA			335			
Escherichia coli DEC5B	WGS	2000	USA	duck		335			
Escherichia coli DEC5C	WGS	1984	Nigeria	chicken		335			
Escherichia coli DEC5D	WGS	1969	Ural			335			
Escherichia coli O55:H7 str. 3256-97	WGS	1952	Canada	Homo sapiens	Gastroenteritis	335		EHEC	
Escherichia coli O55:H7 str. USDA 5905	WGS	1969	Canada	Homo sapiens	Gastroenteritis	335		EHEC	

Figure 19. Screen Shot of Landing Page for Browse Mechanism

5.4.2 Compare mechanism

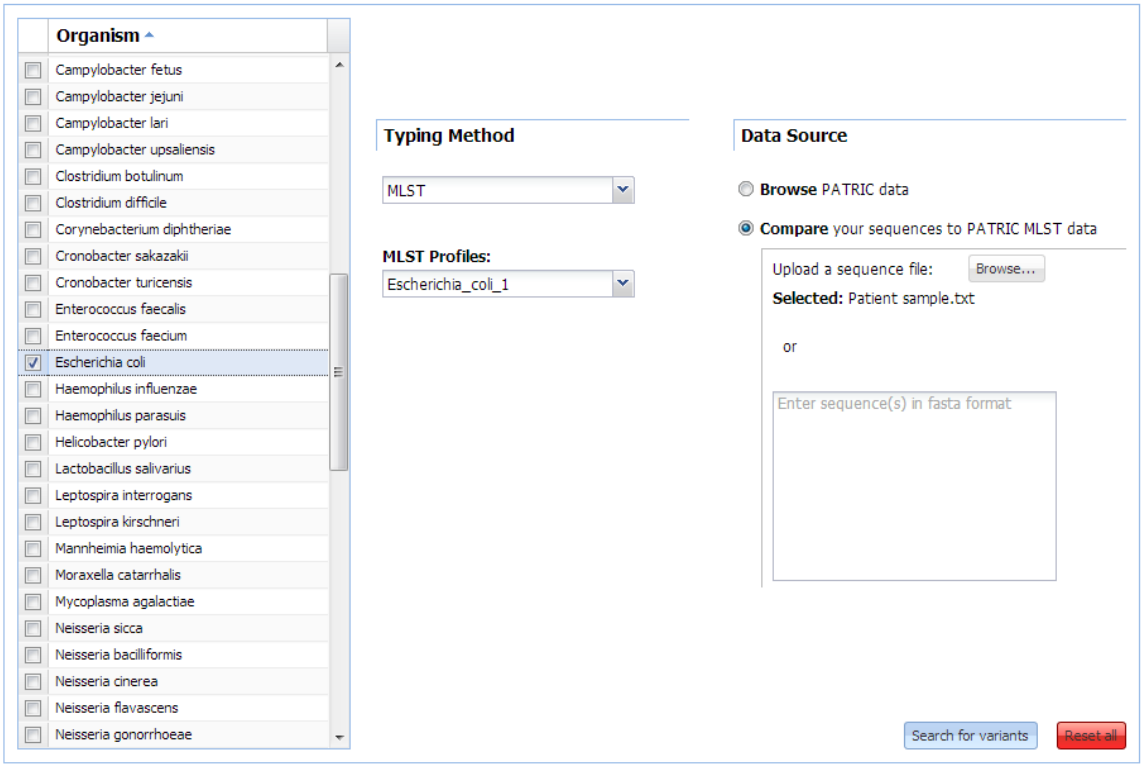


Figure 20. Screen Shot of Search Page for Compare Mechanism

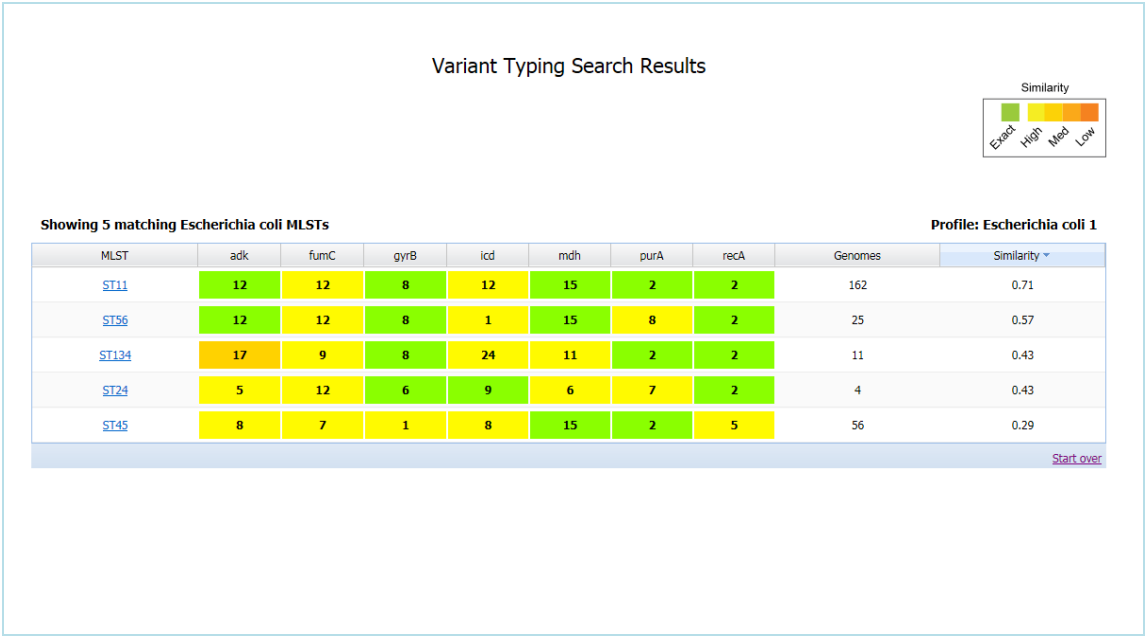


Figure 21. Screen Shot of Results Page for Compare Mechanism

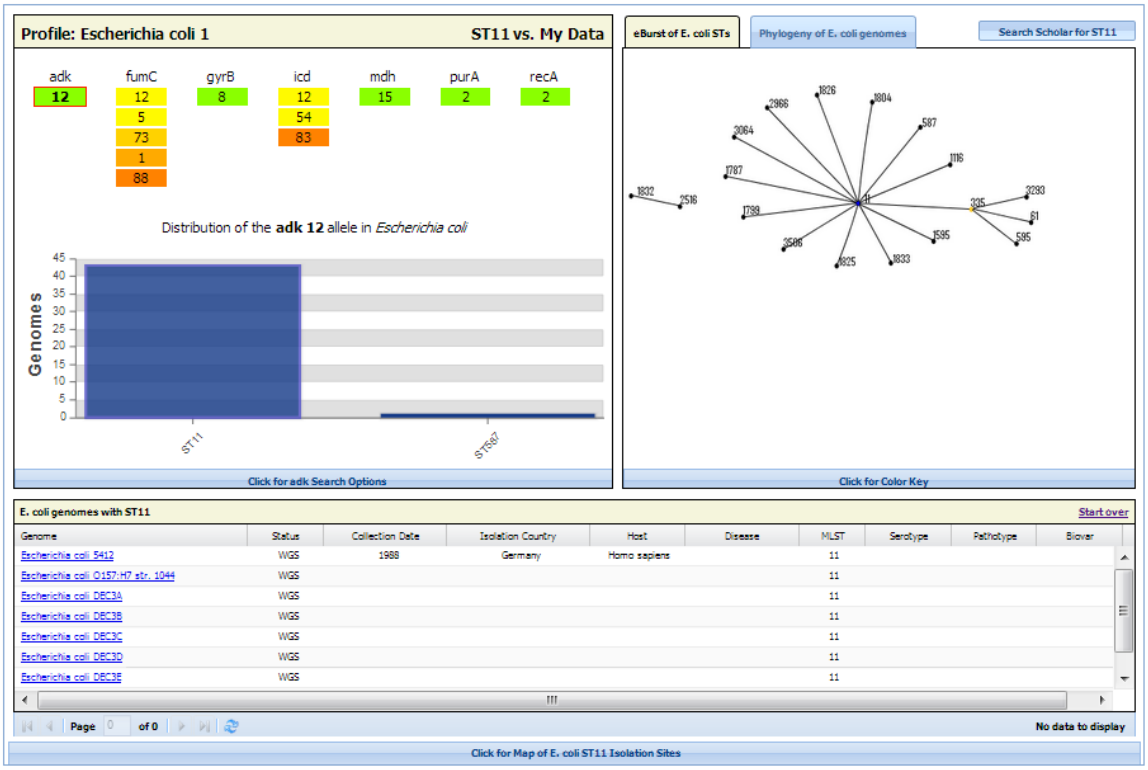


Figure 22. Screen Shot of Landing Page for Compare Mechanism

5.5 Formative Usability Evaluation

Once V2 was live and stable we proceeded to the formative usability evaluation. While the presentation order of benchmark user tasks was structured such that the participant followed a typical workflow, the results are reported by functionally grouping the tasks (i.e., does not necessarily preserve user presentation order). This approach helps bring further insight to each function of the interface as a whole by presenting data from similar components together. The interface is broken down into six main areas of functionality: data input, profile panel, relatives, metadata visualization, report, and outside resources.

Interface Functionality	Description
Data input	Search page
Profile panel	MLST profile broken down into alleles, Bar graph of allele prevalence
Relatives	Phylogeny, eBURST
Metadata visualization	Map depicting where an organism with the MLST profile has been located
Reports	Reports generated based on specific alleles from MLST profile
Outside resources	Link to Google Scholar articles about MLST profile

Figure 23. Interface Functionality Descriptions

We examine each of these six areas using several different metrics including critical incidents, usability insights, content analysis, task efficiency, and subjective ratings. For the purpose of reporting clear and concise results, the critical incidents, usability insights, and content analysis sections will be formatted as bulleted lists. Additionally, critical incidents often reveal usability insights, and therefore there may be some overlap between these categories. Further discussion and analysis can be found in *Section 6.2 Formative Usability Evaluation Analysis.*

5.5.1 Critical incidents

The following graph portrays the frequency of negative critical incidents encountered per task, color-coded and grouped by interface functionality. Critical incidents are counted per participant per task (i.e., two participants who have the same critical incident are counted as two distinct critical incidents in each task) since the prevalence of a particular critical incident is of interest.

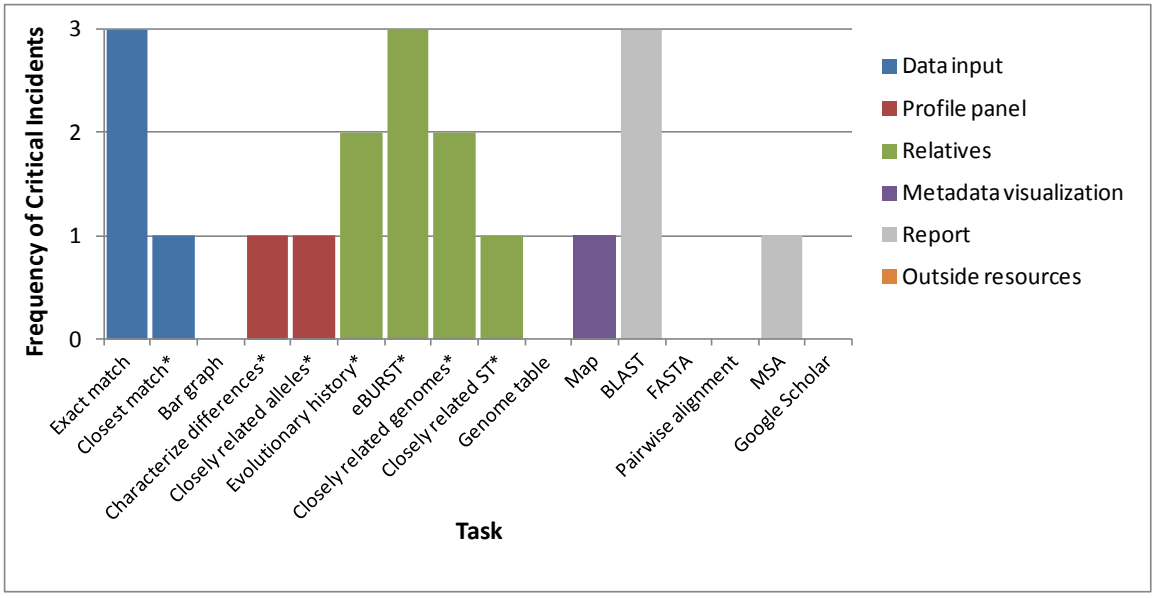


Figure 24. Frequency of Critical Incidents Encountered per Task; Asterisks designate tasks created from participant-developed scenarios

Data input

- 3 participants tried to upload both samples at once
- 1 participant expected to input each allele separately

Profile panel

- 2 participants had difficulty correctly interpreting the colored allele boxes

Relatives

- 3 participants did not find the term "clonal complexes" intuitive
- 2 participants struggled to see the phylogeny as a whole
- 2 participants clicked on the genome table link initially instead of exploring the phylogeny
- 1 participant misinterprets phylogeny legend to mean that brown genomes are a known ST

Metadata visualizations

- 1 participant struggled to find the map functionality

Reports

- 2 participants could not find the reporting options
- 1 participant tried to select the Allele Options button while it was still grayed out (he had not yet selected an allele)

- 1 participant could not find their inputted allele in the MSA report

Outside resources

- No critical incidents

5.5.2 Usability insights

The graph below reveals the frequency of usability insights encountered by task, color-coded and grouped by interface functionality. Usability insights are counted per participant per task (i.e., a usability insight uncovered by two participants in one task is counted as two usability insights), since the prevalence of usability insights is of interest.

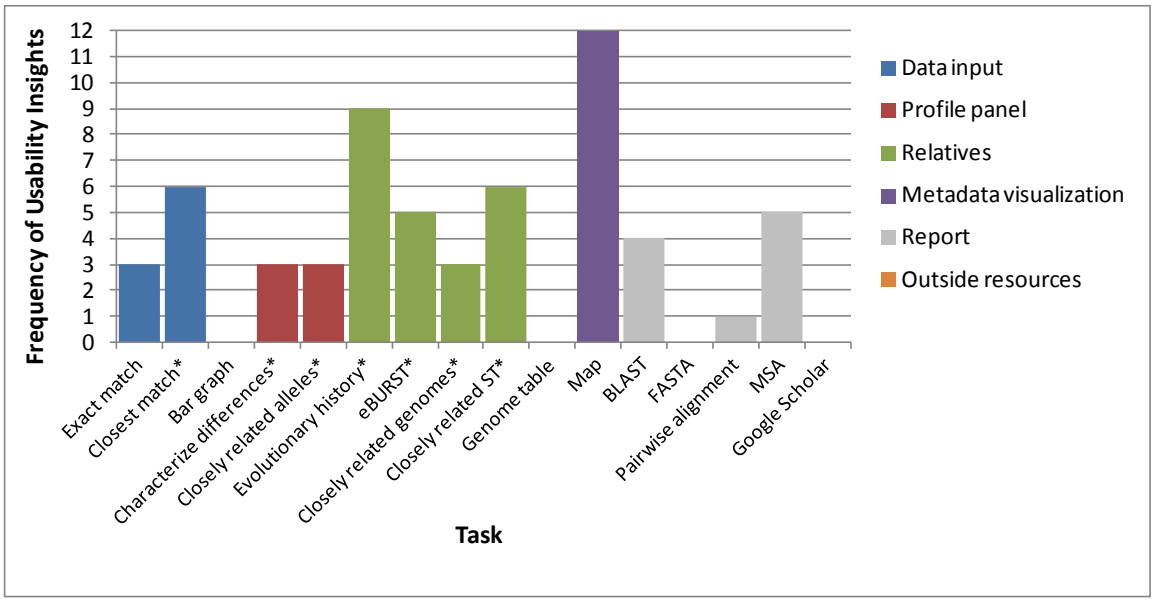


Figure 25. Frequency of Usability Insights Encountered per Task; Asterisks designate tasks created from participant-developed scenarios

Data input

- Include search functionality for the Organism list
- Change "MLST Profiles" text to something more universally recognizable (i.e., MLST Schemes)
- Need batch mode for uploading multiple samples at once
- Include ability to input each allele separately
- Results page should include file names to prevent the user from forgetting which data they are working with

- Include instructions at the top of the Results page
- Change “Matching MLSTs” text to reflect that the results are not exact matches (i.e., Similar MLSTs)
- Include information for how “similarity” is calculated
- Include units for similarity

Profile panel

- Instructions should be included to indicate how to reveal allele mouseovers
- Allele mouseovers should stay visible longer
- Display only the most similar alleles when using the compare mechanism
- Have legend available when using the compare mechanism
- Distinguish the uploaded data in the phylogeny

Relatives

- Include details about how phylogeny is created
- Need ability to view the entire phylogeny at once
- Include metadata as an overlay over the phylogeny
- Phylogeny color key should be more intuitive
- Ability to export phylogeny to phylogeny viewing software
- Include details about how eBURST is created
- eBURST color key should be more intuitive
- Make ST numbers in eBURST more conspicuous
- Make ST colors in eBURST more conspicuous
- Portray single and double locus variants on eBURST

Metadata visualizations

- Make the map button more conspicuous
- Ability to click on a pin and view metadata
- Ability to click on a pin and view contact information
- Ability to click on a pin and jump to strain details page (PATRIC)
- Incorporate ability to view metadata for multiple pins simultaneously
- Include pan and zoom functionality
- Ability to color code pins by ST across an organism

- Incorporate chronological data into map to visually track evolution

Reports

- Make Allele Options functionality more obvious without having to click on it
- Increase conspicuity of Allele Options button
- Include search functionality in reports
- Create more consistent fumC labeling in BLAST report
- Include instructions in Pairwise Alignment report about point mutation notation
- Ability to download MSA
- Increase conspicuity of the inputted allele into the MSA report

Outside resources

- No usability insights

5.5.3 Content analysis

The following data was distilled from detailed notes taken during the formative usability evaluation. General trends and participant comments that were not captured by a critical incident or a usability insight, but were still noteworthy, were collected in a content analysis. This method helps in giving a quick snapshot of users' interaction with the various functionalities of the interface.

Data input

- All participants completed the exact match task, but none of the participants navigated the Search page in exactly the same way
- All participants easily digested the Results page table during the closest match task

Profile panel

- 1 participant commented on his appreciation of the drilldowns
- 2 participants commented on their appreciation of the allele mouseovers
- 1 participant remarked while using the compare mechanism that "to me it raises more questions than answers, but it's always nice to raise questions."
- 1 participant reported while using the compare mechanism that she likes color coding of alleles for a quick visual hint
- 1 participant opined that without metadata the profile panel isn't very useful

- 1 participant stated while using the compare mechanism that the profile panel helps characterize, but does not lead to an answer alone

Relatives

- 3 out of 4 participants used primarily the phylogeny when asked to describe evolutionary history
- 3 participants did not immediately understand the term "clonal complex" meant during task 4
- 2 participants reported finding eBURST "very useful"
- 2 participants clicked first on the genome in the genome table to investigate closely related genomes instead of navigating to the phylogeny
- 3 participants reported wanting to see more of the phylogeny at once
- 2 participants reported wanting to see metadata overlaid on the phylogeny

Metadata visualizations

- 2 participants reported finding the genome table "very useful"
- All participants were able to sort the genome table
- 1 participants wanted to click on entries in the genome table to sort geographically, indicating he had not yet noticed the map functionality
- 2 participants immediately clicked the map button when prompted to find geographical functionality
- All participants commented on how useful they find the map
- 2 participants tried to reconcile genomes in table with map pins
- 2 participants mentioned including a legend for color coding of pins

Reports

- 2 participants needed to be prompted to locate the allele options
- 2 participants went to the pairwise alignment when prompted for the BLAST task
- All participants executed the FASTA task very quickly
- 3 participants reported finding the FASTA "very useful"
- 2 participants commented that the first entry in the MSA should be their inputted allele

Outside resources

- All participants immediately clicked the Scholar button to navigate to Google Scholar
- 1 participant reported that the search criteria were appropriate
- 1 participants opined that he would want more specific search criteria

5.5.4 Task efficiency

The following graph illustrates the mean task time for each task in the usability evaluation. Each task is color-coded and grouped by interface functionality.

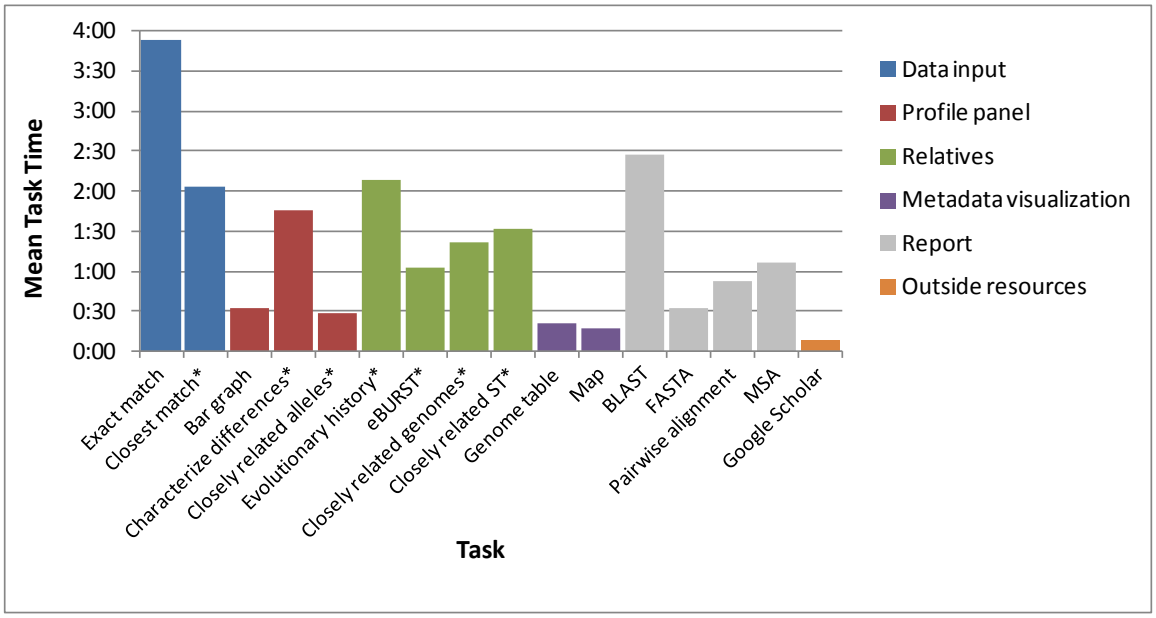


Figure 26. Mean Task Time per Task; Asterisks designate tasks created from participant-developed scenarios

The graph below portrays the task efficiency for each task in the usability evaluation. Efficiency was calculated by dividing task accuracy (i.e., completion rate) by the mean task time.

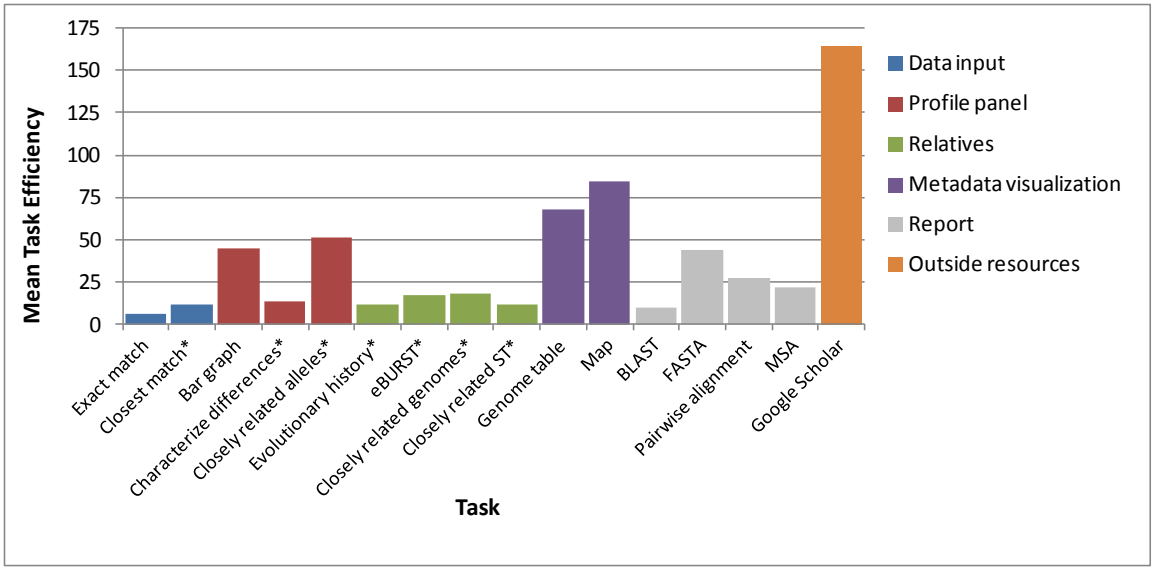


Figure 27. Mean Task Efficiency (Task completion/Mean task time) per Task; Asterisks designate tasks created from participant-developed scenarios

5.5.5 Ratings

The following graph illustrates the mean ease rating for each task in the usability evaluation. Each task is color-coded and grouped by interface functionality.

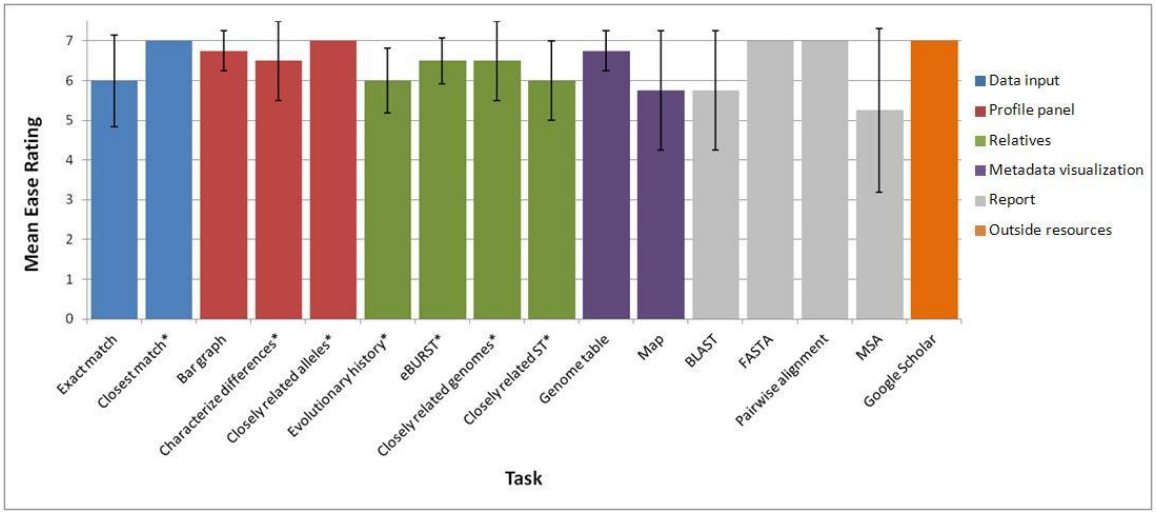


Figure 28. Mean Ease Rating per Task; Asterisks designate tasks created from participant-developed scenarios; Error bars show ± one standard deviation

The graph below shows the mean usefulness rating for each area of functionality of the interface. In some instances, multiple tasks involved the same interface functionality. Therefore, it was natural to combine the usefulness ratings for similar tasks to create an overall usefulness rating for each function.

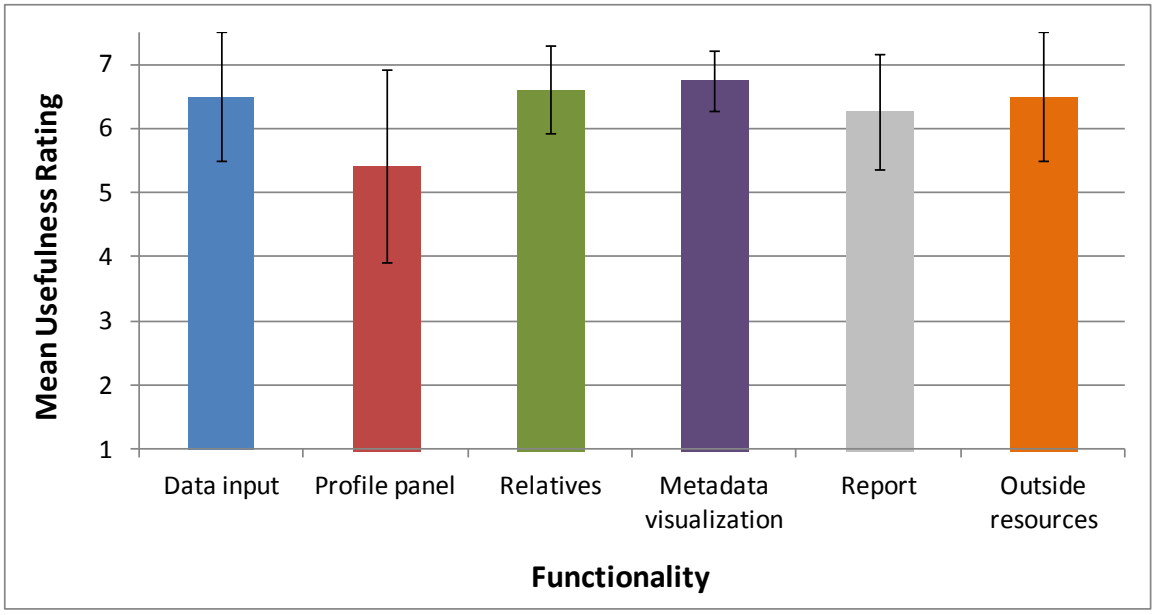


Figure 29. Mean Usefulness Rating per Interface Functionality; Error bars show ± one standard deviation

5.6 Cost-importance Analysis

The cost-importance analysis is sorted from highest to lowest priority. Each usability issue was rated on a scale of one to five for both importance and cost. The priority of each usability issue was determined using the following formula:

Priority = Importance / Cost

All usability issues with a priority of one or higher were given to the software developer to develop V3. Implementing solutions to the usability issues, and the resulting V3 MLST tool generated from these implementations are beyond the scope of this research.

Table 4. Cost-importance Analysis.

Interface Functionality	Usability Solution	Importance Rating	Cost Rating	Priority
Reports	Increase conspicuity of Allele Options button	5	1	5.00
Relatives	Include details about how eBURST is created	4	1	4.00
Data input	Include uploaded data file names on Results page	3	1	3.00

Data input	Include information for how “similarity” is calculated	3	1	3.00
Reports	Include search functionality in reports	3	1	3.00
Reports	Make Allele Options functionality more obvious without having to click on it	5	2	2.50
Data input	Include instructions at the top of the Results page	2	1	2.00
Data input	Change “Matching MLSTs” text to reflect that the results are not exact matches	2	1	2.00
Relatives	Include details about how phylogeny is created	2	1	2.00
Metadata visualization	Make the map button more conspicuous	4	2	2.00
Reports	Increase conspicuity of the inputted allele into the MSA report	4	2	2.00
Data input	Incorporate batch mode for uploading multiple samples at once	3	2	1.50
Data input	Change “MLST Profiles” text to something more universally recognizable	1	1	1.00
Data input	Include units for similarity	1	1	1.00
Profile panel	Instructions should be included to indicate how to reveal allele mouseovers	2	2	1.00
Profile panel	Display only the most similar alleles when using the compare mechanism	3	3	1.00
Profile panel	Have legend available when using the compare mechanism	1	1	1.00
Relatives	Ability to export phylogeny to phylogeny viewing software	2	2	1.00
Metadata visualization	Ability to click on a pin and view metadata	5	5	1.00
Metadata visualization	Include pan and zoom functionality	5	5	1.00
Reports	Ability to download MSA	2	2	1.00
Profile panel	Allele mouseovers should stay visible longer	3	4	0.75
Relatives	Incorporate ability to view the entire phylogeny at once	3	4	0.75

Profile panel	Distinguish the uploaded data in the phylogeny	2	3	0.67
Data input	Include ability to input each allele separately	1	2	0.50
Reports	Include instructions in Pairwise Alignment report about point mutation notation	1	2	0.50
Relatives	Include metadata as an overlay over the phylogeny	2	5	0.40
Relatives	Portray single and double locus variants on eBURST	2	5	0.40
Metadata visualization	Ability to click on a pin and jump to strain details page (PATRIC)	2	5	0.40
Metadata visualization	Incorporate ability to view metadata for multiple pins simultaneously	2	5	0.40
Data input	Include search functionality for the Organism list	1	3	0.33
Relatives	Phylogeny color key should be more intuitive	1	3	0.33
Relatives	eBURST color key should be more intuitive	1	3	0.33
Reports	Create more consistent fumC labeling in BLAST report	1	3	0.33
Relatives	Make ST numbers in eBURST more conspicuous	1	5	0.20
Relatives	Make ST colors in eBURST more conspicuous	1	5	0.20
Metadata visualization	Ability to click on a pin and view contact information	1	5	0.20
Metadata visualization	Ability to color code pins by ST across an organism	1	5	0.20
Metadata visualization	Incorporate chronological data into map in order to visually track evolution	1	5	0.20

CHAPTER 6. DISCUSSION

6.1 Requirements Analysis

Resulting requirements from the scenario-based interviews suggest that a less structured scenario-based interview scheme yields not only a greater quantity of requirements, but also produced requirements that are more creative.

6.1.1 Requirements within epidemiologist user class

To address *Research Question 1*, the requirements elicited from each of the three levels of seed scenarios within the epidemiologist user class were organized into three separate groups. It is important to note that of the 14 requirements generated from the scenario-based interviews, only seven came from the structured scenario portion of the interview.

The findings suggest that as scenarios move away from extant work practices they actually yield less creative requirements, therefore refuting *Hypothesis 1*. Of requirements generated from seed scenarios, the most requirements came from the commonplace scenario, and the least requirements came from the radical scenario. Proposing scenarios that are beyond users' current work practices seemed to overwhelm and confuse participants, and therefore unexpectedly worked against requirements generation to the point where the participant could not conceive of a basic way to extend their work practices. In the structured portion of the interview, participants seemed to want more of the same of what they are already doing with MLST. One epidemiologist could think beyond current practices in terms of what would be interesting but did not know how to execute it, while the other epidemiologist is unsure of how to extend his work practices.

Table 5. Number of Requirements Generated at Each Level of Seed Scenario

	Number of Requirements
Scenario 1 (common)	5
Scenario 2 (moderate)	2
Scenario 3 (radical)	0

The difference in requirements as scenarios move further from extant work practices was not so much a difference in quality (i.e., degree of creativity), but a difference in quantity. While the quality of requirements is certainly important, quality is only relevant when there is a substantial quantity. Presenting radical seed scenarios to participants did not seem to provoke insight and creativity, but rather seemed overwhelming. It is plausible that the scenarios were not given enough context, or that not enough preparations was done to get the participant in the right frame of mind to become inspired about answering questions beyond their current work practices.

While the purpose of the epidemiologist interviews was to understand requirements at each level of seed scenario, it was natural to speak more freely with participants during follow-up questions when following the flow of conversation. Seven of the epidemiologist requirements resulted from this non-scenario portion of the interviews. During the non-scenario portion of the interview, relevant requirements emerged on the level of the moderately radical scenario. The generation of seven additional requirements from the non-scenario portion of the interviews demonstrates the participants' alternate state of mind when speaking more openly. This more natural style of discussion seemed to be more conducive to free and imaginative thinking. Though the non-scenario portion of the interview did not generate any extreme requirements, participants did move beyond extant work practices.

6.1.2 Requirements between user classes

To address *Research Question 2*, the requirements elicited from the structured scenarios (epidemiologists) and the free-form scenarios (molecular biologists) were

organized into two separate groups. The structured scenarios yielded seven requirements, and the free-form scenarios yielded 18 requirements. All seven requirements from the structured scenarios were also elicited from the free-form scenarios; that is, these seven were identified via both the structured and free-form scenario approaches.

The common requirements between the groups were more high-level and general. This type of requirement is still useful and necessary, but it does not provide great insight into the form of a next-generation MLST tool. Five of these requirements can be attributed to current work practices, but the remaining two requirements, while certainly extending beyond current practices, are merely alluding to high-level functionality without parsing out further detail. However, this could provide a springboard for additional follow-on discussion to extend the capability of users' workflow.

The findings suggest that elicited requirements from structured and free-form scenario groups are actually quite different with respect to range of MLST functionality and level of creativity, therefore refuting *Hypothesis 2*. As discussed in the previous section, epidemiologists seemed to be confined and mentally limited by their scenarios. Molecular biologists, on the other hand, seemed to thrive with their open-ended scenarios and gentle probe questions. While moving further from extant work practices with seed scenarios caused confusion to epidemiologists and did not result in next-generation level requirements, requirements elicited from molecular biologists were heavily geared towards extended work practices. This outcome is potentially due to the nature of the work of each of the user classes; epidemiologists typically have a formulaic rule-based method of solving problems with MLST, while molecular biologists use MLST to do research that is more knowledge-based. The more imaginative work of molecular biologists may be linked to a greater capacity for creative thinking pertaining to MLST. Four requirements elicited from molecular biologists are attributed to current work practices, while the remaining 12 are attributed to extended work practices.

6.2 Formative Usability Evaluation Analysis

Participants responded favorably to the structured interaction with the MLST tool as suggested by the overall qualitative results of the formative usability evaluation. The interface was robust enough to support participants' successful completion of the vast

majority of tasks, yet still offered a useful medium for elicitation of usability insights. Usability themes converged across participants, further substantiating the claim established in *Section 4.2 Participants* that the four-participant sample is sufficient.

6.2.1 General findings

Critical incidents were most often encountered during the first task of each the data input and reports functions. The eBURST task also yielded a high number of critical incidents, but this is arguably due to poor wording of the task prompt. These findings are unsurprising, given that participants would logically struggle most when first encountering a specific function of an interface. The reduction in critical incidents after the first task of these functions suggests that participants were quickly learning how to better interact with the interface. Almost all of the encountered critical incidents translate to relevant usability insights. Notably, not all critical incidents were leveraged for usability insights due to impracticality or failure to align with the ”next generation” direction intended for this MLST tool.

Tasks that generated the most usability insights were, in general, tasks that were the furthest departure from a traditional MLST tool: investigating the closest match to specific genome data, utilizing eBURST and a phylogeny simultaneously to explore evolutionary history, and examining a map. Given that participants already had preexisting mental models for genome tables and FASTA reports, it is unsurprising that these tasks were less challenging to navigate. An outlier to this trend was the bar graph task, which is new functionality that participants did not encounter any difficulty interacting with and interpreting. While it does take some time to build a mental model of new functionality, the interface needs to afford easy processing of new information. The resulting usability insights drawn forth from this evaluation will help to facilitate this process.

Participants had many positive interactions with the various functions of the interface, as is evident from the content analysis. Participants each interacted with the interface in their own way; whether it was executing an open-ended task in a different order, or manipulating specific functionality with their own style. There was no

significant difference in interaction based on user class, as indicated by comparable critical incident, usability insight, and task efficiency data collected from each user class.

Resulting task efficiencies did not yield any surprises; tasks that were less complex and more familiar were most efficient, and tasks that were more complex and less familiar were least efficient. Only two tasks, eBURST and Closely related ST, were not completed by all participants, indicating that task completion did not have a large impact on task efficiency.

Overall, participants did not find the tasks overly challenging, judging by the content analysis and ease rating for each task. Although the ease ratings are not dissimilar enough to infer any trends between tasks, the fact that most tasks means were a six or above suggests a general trend of ease when interacting with the interface. Usefulness ratings exhibit similar characteristics, where there are not wide differences between interface functionality, but five out of six of the areas of functionality were rating greater than a six in usefulness.

6.2.2 Participant-developed scenario findings

Seven of the 16 tasks were created from participant-developed scenarios in the first phase of this research. These seven tasks encompassed three of the main areas of functionality of the interface: data input, profile panel, and relatives. This functionality align with the nature of the work of molecular biologists, which constituted the user class participating in the open-ended scenario exercise.

All participant-inspired tasks generated both critical incidents and usability insights, unlike some other tasks that did not provoke either. While critical incidents do reflect a participant facing a challenge with that task, there is something to be said for evoking a reaction to a component of the interface at all. Given that the nature of the applied component of this work is to inspire a next-generation MLST tool, it should not be surprising that participants are not as accustomed to interacting with the types of tasks inspired from participant-developed scenarios. As mentioned previously, critical incidents and usability insights often go hand in hand; therefore, participant-inspired tasks also generated a considerable amount of valuable usability data.

Table 6. Mean Critical Incidents and Usability Insights by Task Inspiration

	Participant-inspired Tasks	Not Participant-inspired Tasks
Mean Critical Incidents	1.57	0.89
Mean Usability Insights	4.14	2.11

Participant-inspired tasks generated more critical incidents overall as compared to those not inspired by participants. The critical incidents encountered from these tasks were not so severe that the participant could not complete the task, they simply provided a useful insight into participant interaction with the interface, and revealed opportunities for improvement.

Participant-inspired tasks generated a substantial body of rich usability data. The Relatives function was created entirely from participant-developed scenarios, and uncovered many usability insights from both user classes. The more complex task from the Data input function was also created from a participant-developed scenario, in addition to the subsequent tasks from the Profile panel function. Participants may have encountered similar functionality as utilized in these tasks elsewhere, potentially combining the benefit of some prior knowledge with the unfamiliarity of working with many components simultaneously. This combination of a general familiarity with a component, along with interacting with that component in the usability evaluation as part of a greater tool, proved to be advantageous in revealing usability insights. Another advantageous mix captured in this research was the dichotomy between the two user classes. Since the molecular biologist user class essentially created some of the tasks, and the epidemiologist user class was not involved, these two experience levels theoretically demonstrated both a novice and expert perspective in the formative usability evaluation. Epidemiologists are certainly not true novices in the field of MLST, but they are likely less experienced in general with the more heavily research-based tools used by molecular biologists.

The findings support *Hypothesis* 3, namely that participant-inspired tasks were ultimately effective as benchmark tasks. The participant-inspired tasks were intricate enough to afford meaningful interaction with the interface, while still being completable by both user classes. Essentially, the participant-inspired tasks were useful as benchmark

tasks because they resulted in useful usability data. Tasks were plentiful enough to be organized into task flows, while also demonstrating a range of difficulty. Though tasks inspired by participants yielded substantial usability data, the tasks did not encompass the whole range of interactions a user might have with the interface. In this sense, participant-inspired tasks were a useful starting point for benchmark tasks, but should not be relied on primarily to span the breadth of interactions.

6.3.3 Situation awareness findings

Thought situation awareness (SA) was not the focus of this research, there are interesting links to be explored between this work and SA. The MLST tool presented to participants during the formative usability evaluation demonstrated a cohesive system combining many relevant features typically sought out in various locations. Participants' enthusiasm for this next-generation cohesive system was evident by their eagerness to explore and interact with each component of the interface. Multiple participants used information from two components in unison to gain deeper SA than if the components were separate. Participants also remarked about the convenience of having specific functionality, that they would typically need to seek out separately in another system, immediately available to them. Participants did not appear to find the interface perceptually overwhelming, as evidenced by their swift interactions. Additionally, the interface seemed to reduce the cognitive load put on working memory by providing a single visualization containing a span of functionality, and by aligning with participants' mental models.

CHAPTER 7. CONCLUSIONS

Though scenarios have become a common tool, more advanced scenario methodologies have yet to be fully explored. This research helps to provide a greater understanding of the utilization of novel scenario styles and methodologies.

This research generated data pertaining to levels of scenarios, difference in scenario makeup, and the use of scenarios throughout the usability engineering lifecycle. The different levels of seed scenarios presented to participants demonstrate how departure from extant work practices paired with a lack of context seemed to stifle creativity. On the other hand, the open-ended scenario style pushed participants into more creative and original thought. The open-ended scenario style also proved successful in usability testing by serving as effective benchmark tasks that generated significant usability data.

Though the original intention of seed scenarios was to inspire creativity when moving away from extant work practices, the findings still serve the greater body of knowledge on scenarios by investigating new scenario styles and methodologies. Additionally, this research generated promising data regarding the use of the open-ended scenario style for next-generation level requirements elicitation, and as benchmark tasks for usability testing.

This scenario research is limited by the MLST interface and the participants' scope of work. Though MLST provided a useful platform for scenario investigation, much more can be done by leveraging other fields and next-generation applications. Future work should explore a similar scenario methodology applied not just to bioinformatics, but a wide array of fields. Future work should also investigate other methodologies with the open-ended scenario style throughout the usability engineering lifecycle. While this work reveals the potential of such scenarios, further investigation of various methodologies is required for effective utilization.

The results of this study provide support for the continued investigation into novel uses of scenarios for a variety of applications. The methodology used in this research for levels of scenarios and difference in scenario style provide a platform for the use of scenarios in yielding creative requirements, and can be applied to a multitude of fields.

The scenarios have been shown to fare well when continuing from requirements elicitation into usability testing, proving the benefit of incorporating scenarios into the usability engineering lifecycle. This specific application reveals the vast flexibility and potential of scenarios yet to be fully explored.

REFERENCES

Antón, A. I., & Potts, C. (1998). A representational framework for scenarios of system use. *Requirements Engineering, 3*(3), 219-241.

Carroll, J. M. (1995). *Scenario-based design: Envisioning work and technology in system development*. New York, NY: John Wiley & Sons.

Carroll, J. M. (2000). *Making use: Scenario-based design of human-computer interactions*. Cambridge, MA: MIT Press.

Carroll, J. M. (2002). Making use is more than a matter of task analysis. *Interacting with Computers, 14*(5), 619-628.

Chin, G., Jr. (2004). *A case study in the participatory design of a collaborative science-based learning environment.* Virginia Polytechnic Institute and State University.

Durso, F. T., & Gronlund, S. D. (1999). Situation awareness. In F. T. Durso (Ed.), *Handbook of applied cognition* (pp. 283-314). Chichester, UK: John Wiley & Sons Ltd. .

Durso, F. T., & Sethumadhavan, A. (2008). Situation awareness: Understanding dynamic environments. *Human Factors: The Journal of the Human Factors and Ergonomics Society, 50*(3), 442-448.

Endsley, M. R. (1988). *Design and evaluation for situation awareness enhancement.* Proceedings from Human Factors and Ergonomics Society Annual Meeting. Santa Monica, CA: SAGE Publications.

Endsley, M. R. (1995). Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society, 37*(1), 32-64.

Endsley, M. R. (2003). *Designing for situation awareness: An approach to user-centered design*. New York, NY: CRC Press.

Ericsson, K. A., & Simon, H. A. (1998). How to study thinking in everyday life: Contrasting think-aloud protocols with descriptions and explanations of thinking. *Mind, Culture, and Activity, 5*(3), 178-186.

Flanagan, J. C. (1954). The critical incident technique. *Psychological Bulletin, 51*(4), 327-357.

Gaba, D. M., Howard, S. K., & Small, S. D. (1995). Situation awareness in anesthesiology. *Human Factors: The Journal of the Human Factors and Ergonomics Society, 37*(1), 20-31.

Gentner, D., & Stevens, A. L. (1983). *Mental models*. Hillsdale, NJ: Erlbaum.

Glinz, M. (2000). *Improving the quality of requirements with scenarios.* Proceedings from Second World Congress for Software Quality. Yokohama, Japan.

Gould, J. D., & Lewis, C. (1985). Designing for usability: Key principles and what designers think. *Communications of the ACM, 28*(3), 300-311.

Gugerty, L. J. (1997). Situation awareness during driving: Explicit and implicit knowledge in dynamic spatial memory. *Journal of Experimental Psychology Applied, 3*(1), 42-66.

Hartson, H. R. (1998). Human–computer interaction: Interdisciplinary roots and trends. *Journal of Systems and Software, 43*(2), 103-118.

Hartson, H. R., & Pyla, P. S. (2012). *The UX book: Process and guidelines for ensuring a quality user experience*. San Francisco, CA: Morgan Kaufmann.

Hoeft, R. M., & Mentis, H. M. (2009). *Beyond user-centered design: Applicable concepts from complementary approaches.* Proceedings from Human Factors and Ergonomics Society Annual Meeting. San Antonio, TX: SAGE Publications.

Hsia, P., Samuel, J., Gao, J., Kung, D., Toyoshima, Y., & Chen, C. (1994). Formal approach to scenario analysis. *IEEE Software, 11*(2), 33-41.

ISO. (1998). 9241-11. Ergonomic requirements for office work with visual display terminals (VDTs). *The international organization for standardization*.

ISO. (1999). 13407: Human-centred design processes for interactive systems. *Geneva: ISO*.

Köser, C. U., Ellington, M. J., Cartwright, E. J. P., Gillespie, S. H., Brown, N. M., Farrington, M., . . . Parkhill, J. (2012). Routine use of microbial whole genome sequencing in diagnostic and public health microbiology. *PLoS Pathogens, 8*(8), e1002824.

Kuutti, K. (1995). Work processes: Scenarios as a preliminary vocabulary. In J. M. Carroll (Ed.), *Scenario-based design: Envisioning work and technology in system development* (pp. 19-36): John Wiley & Sons.

Laporti, V., Borges, M. R. S., & Braganholo, V. (2009). Athena: A collaborative approach to requirements elicitation. *Computers in Industry, 60*(6), 367-380.

Liu, Z. L., Zhang, Z., & Chen, Y. (2012). A scenario-based approach for requirements management in engineering design. *Concurrent Engineering, 20*(2), 99-109.

Maiden, M. C. J., Bygraves, J. A., Feil, E., Morelli, G., Russell, J. E., Urwin, R., . . . Caugant, D. A. (1998). Multilocus sequence typing: A portable approach to the identification of clones within populations of pathogenic microorganisms. *Proceedings of the National Academy of Sciences, 95*(6), 3140-3145.

Mao, J. Y., Vredenburg, K., Smith, P. W., & Carey, T. (2005). The state of user-centered design practice. *Communications of the ACM, 48*(3), 105-109.

Neale, D. C., & Kies, J. K. (1996). *Scenario-based design for human-computer interface development.* Proceedings from Human Factors and Ergonomics Society Annual Meeting. Philadelphia, PA: SAGE Publications.

Nickerson, R., & Landauer, T. (1997). Human-computer interaction: Background and issues. In M. G. Helander, T. K. Landauer & P. Prabhu (Eds.), *Handbook of human-computer interaction*. Amsterdam, The Netherlands: Elsevier Science.

Nielsen, J. (1994). *Usability engineering*. San Diego, CA: Academic Press.

Potts, C., Takahashi, K., & Anton, A. I. (1994). Inquiry-based requirements analysis. *IEEE Software, 11*(2), 21-32.

Rasmussen, J. (1983). Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-13*(3), 257-266.

Rolland, C., Ben Achour, C., Cauvet, C., Ralyté, J., Sutcliffe, A., Maiden, N., . . . Dubois, E. (1998). A proposal for a scenario classification framework. *Requirements Engineering, 3*(1), 23-47.

Rosson, M. B., & Carroll, J. M. (1995). Narrowing the specification-implementation gap in scenario-based design. In J. M. Carroll (Ed.), *Scenario-based design: Envisioning work and technology in system development* (pp. 247-278). New York, NY: Wiley & Sons.

Rosson, M. B., & Carroll, J. M. (2002). Scenario-Based Design. In J. A. Jacko & A. Sears (Eds.), *The human-computer interaction handbook: Fundamentals, evolving*

technologies and emerging applications (pp. 1032-1050). Mahwah, NJ: Lawrence Erlbaum Associates.

Stanton, N. A., Stewart, R., Harris, D., Houghton, R. J., Baber, C., McMaster, R., . . . Young, M. S. (2006). Distributed situation awareness in dynamic systems: Theoretical development and application of an ergonomics methodology. *Ergonomics, 49*(12-13), 1288-1311.

Sutcliffe, A. G. (2003). *Scenario-based requirements engineering.* Proceedings from RE '03. Monterey Bay, CA: IEEE.

Sutcliffe, A. G., Maiden, N. A. M., Minocha, S., & Manuel, D. (1998). Supporting scenario-based requirements engineering. *IEEE Transactions on Software Engineering, 24*(12), 1072-1088.

Terrell, I. S., McNeese, M. D., & Jefferson, T. (2004). *Exploring cognitive work within a 911 dispatch center: Using complementary knowledge elicitation techniques.* Proceedings from Human Factors and Ergonomics Society Annual Meeting. New Orleans, LA: SAGE Publications.

Virzi, R. A. (1992). Refining the test phase of usability evaluation: How many subjects is enough? *Human Factors: The Journal of the Human Factors and Ergonomics Society, 34*(4), 457-468.

Weidenhaupt, K., Pohl, K., Jarke, M., & Haumer, P. (1998). Scenarios in system development: Current practice. *IEEE Software, 15*(2), 34-45.

Wixon, D., & Wilson, C. (1997). The usability engineering framework for product design and evaluation. In M. G. Helander, T. K. Landauer & P. Prabhu (Eds.), *Handbook of human-computer interaction* (pp. 653-688). Amsterdam, The Netherlands: Elsevier Science.

Appendix A: Virginia Tech IRB Approval

A.1 IRB Approval

VirginiaTech

Office of Research Compliance
Institutional Review Board
2000 Kraft Drive, Suite 2000 (0497)
Blacksburg, VA 24060
540/231-4606 Fax 540/231-0959
email irb@vt.edu
website http://www.irb.vt.edu

MEMORANDUM

DATE: October 19, 2012

TO: Joseph L Gabbard Jr, Stephanie Alpert, Meredith Jane Corielle Wilson

FROM: Virginia Tech Institutional Review Board (FWA00000572, expires May 31, 2014)

PROTOCOL TITLE: User-centered Development of MLST Tool and Related Concepts for PATRIC

IRB NUMBER: **12-881**

Effective October 18, 2012, the Virginia Tech Institution Review Board (IRB) Chair, David M Moore, approved the New Application request for the above-mentioned research protocol.

This approval provides permission to begin the human subject activities outlined in the IRB-approved protocol and supporting documents.

Plans to deviate from the approved protocol and/or supporting documents must be submitted to the IRB as an amendment request and approved by the IRB prior to the implementation of any changes, regardless of how minor, except where necessary to eliminate apparent immediate hazards to the subjects. Report within 5 business days to the IRB any injuries or other unanticipated or adverse events involving risks or harms to human research subjects or others.

All investigators (listed above) are required to comply with the researcher requirements outlined at:

http://www.irb.vt.edu/pages/responsibilities.htm

(Please review responsibilities before the commencement of your research.)

PROTOCOL INFORMATION:

Approved As:	**Expedited, under 45 CFR 46.110 category(ies) 7**
Protocol Approval Date:	**October 18, 2012**
Protocol Expiration Date:	**October 17, 2013**
Continuing Review Due Date*:	**October 3, 2013**

*Date a Continuing Review application is due to the IRB office if human subject activities covered under this protocol, including data analysis, are to continue beyond the Protocol Expiration Date.

FEDERALLY FUNDED RESEARCH REQUIREMENTS:

Per federal regulations, 45 CFR 46.103(f), the IRB is required to compare all federally funded grant proposals/work statements to the IRB protocol(s) which cover the human research activities included in the proposal / work statement before funds are released. Note that this requirement does not apply to Exempt and Interim IRB protocols, or grants for which VT is not the primary awardee.

The table on the following page indicates whether grant proposals are related to this IRB protocol, and which of the listed proposals, if any, have been compared to this IRB protocol, if required.

Date*	OSP Number	Sponsor	Grant Comparison Conducted?
10/17/2012	09140101	National Institute of Allergy & Infectious Diseases	Compared on 10/17/2012

* Date this proposal number was compared, assessed as not requiring comparison, or comparison information was revised.

If this IRB protocol is to cover any other grant proposals, please contact the IRB office (irbadmin@vt.edu) immediately.

A.2 IRB Re-approval

VirginiaTech

Office of Research Compliance
Institutational Review Board
North End Center, Suite 4120, Virginia Tech
300 Turner Street NW
Blacksburg, Virginia 24061
540/231-4606 Fax 540/231-0959
email irb@vt.edu
website http://www.irb.vt.edu

MEMORANDUM

DATE:	September 19, 2013
TO:	Joseph L Gabbard Jr, Stephanie Alpert, Meredith Jane Corielle Wilson
FROM:	Virginia Tech Institutional Review Board (FWA00000572, expires April 25, 2018)
PROTOCOL TITLE:	User-centered Development of MLST Tool and Related Concepts for PATRIC
IRB NUMBER:	**12-881**

Effective September 19, 2013, the Virginia Tech Institution Review Board (IRB) Chair, David M Moore, approved the Continuing Review request for the above-mentioned research protocol.

This approval provides permission to begin the human subject activities outlined in the IRB-approved protocol and supporting documents.

Plans to deviate from the approved protocol and/or supporting documents must be submitted to the IRB as an amendment request and approved by the IRB prior to the implementation of any changes, regardless of how minor, except where necessary to eliminate apparent immediate hazards to the subjects. Report within 5 business days to the IRB any injuries or other unanticipated or adverse events involving risks or harms to human research subjects or others.

All investigators (listed above) are required to comply with the researcher requirements outlined at:

http://www.irb.vt.edu/pages/responsibilities.htm

(Please review responsibilities before the commencement of your research.)

PROTOCOL INFORMATION:

Approved As:	**Expedited, under 45 CFR 46.110 category(ies) 7**
Protocol Approval Date:	**October 18, 2013**
Protocol Expiration Date:	**October 17, 2014**
Continuing Review Due Date*:	**October 3, 2014**

*Date a Continuing Review application is due to the IRB office if human subject activities covered under this protocol, including data analysis, are to continue beyond the Protocol Expiration Date.

FEDERALLY FUNDED RESEARCH REQUIREMENTS:

Per federal regulations, 45 CFR 46.103(f), the IRB is required to compare all federally funded grant proposals/work statements to the IRB protocol(s) which cover the human research activities included in the proposal / work statement before funds are released. Note that this requirement does not apply to Exempt and Interim IRB protocols, or grants for which VT is not the primary awardee.

The table on the following page indicates whether grant proposals are related to this IRB protocol, and which of the listed proposals, if any, have been compared to this IRB protocol, if required.

Date*	OSP Number	Sponsor	Grant Comparison Conducted?
10/17/2012	09140101	National Institute of Allergy & Infectious Diseases	Compared on 10/17/2012

* Date this proposal number was compared, assessed as not requiring comparison, or comparison information was revised.

If this IRB protocol is to cover any other grant proposals, please contact the IRB office (irbadmin@vt.edu) immediately.

Appendix B: Scenario-based Interview Protocols

B.1 Epidemiologist Interview Protocol

INTRODUCTION

Thank you for electing to participate in this study. As we have mentioned, we are looking to gather information to inform the design of a next generation MLST system. I (Meredith Wilson) am a Biosciences Analyst with a background in epidemiology at the Virginia Bioinformatics Institute (VBI) at Virginia Tech. I am here with Joe Gabbard, a Research faculty with expertise in human-computer interaction (HCI) and user experience (UX), and Stephanie Alpert, a Graduate student doing R&D for our PATRIC bioinformatics website here at VBI.

Today we are going to talk for about 45 minutes about your common work practices and domain knowledge. We will start with a few background questions, and then work through a few "scenario" activities that I will explain in more detail later. I am now going to hand the conversation over to Stephanie, who will ask you some background information.

DEMOGRAPHIC INFORMATION

- What is your age?

- What field(s) are you trained in, what is your occupation?

- How long have you been working in your field?

BACKGROUND INFORMATION

- What kinds of subtyping methods are you most familiar with? What do you like or dislike about these methods?

- Do you work with any infectious agents? Which ones?

SCENARIOS

I am going to introduce to you three different hypothetical scenarios that are common to the field of infectious disease surveillance. In each of the scenarios, I will ask you to pretend that you are an epidemiologist working at a county health department and I am your boss. I will read the scenario to you, and ask you to describe how you would complete the task. If this is a new process to you, that is okay, I am here to answer any questions you have if you need guidance, just as if I was your real boss.

"One day I come to you with a bunch of notifications from a local hospital for patients that are presenting with gastroenteritis (abdominal pain, diarrhea, vomiting...). The patients are all 18 - 25 years old from the same university, and are seropositive for Escherichia coli. Despite whether or not we have used it before, I tell you that the CDC is now requiring us to use MLST sequence types as our subtyping method, so this is what you will use for molecular source attribution.

You receive individual PCR sequencing of MLST loci extracted from the patient samples, and the sequence data for the individual genes. You also get samples and corresponding MLSTs Escherichia coli from suspected sources in the university dining hall (*in scenario 2, interviewee gets whole genomes rather than individual gene sequences*). Now that you have all of your individual MLST gene loci, what would you do?"

Scenario 1: STs from patients perfect match to chicken

Scenario 2: (get whole genomes rather than individual loci) Sequence type from patients perfect match to chicken

Scenario 3: STs from patients and sources are all of the same sequence type

Scenario 4: Could you walk us through a challenging case that you cracked?

Follow-up Questions

- Did you find this scenario easy or difficult?
- What was easy or difficult about it?

B.2 Molecular Biologist Interview Protocol

INTRODUCTION

Thank you for electing to participate in this study. As we have mentioned, we are looking to gather information to inform the design of a next generation MLST system. I (Meredith Wilson) am a Biosciences Analyst with a background in epidemiology at the Virginia Bioinformatics Institute (VBI) at Virginia Tech. I am here with Joe Gabbard, a Research faculty with expertise in human-computer interaction (HCI) and user experience (UX), and Stephanie Alpert, a Graduate student doing R&D for our PATRIC bioinformatics website here at VBI.

Today we are going to talk for about 45 minutes about your common work practices and domain knowledge. We will start with a few background questions, and then work through a few "scenario" activities that I will explain in more detail later. I am now going to hand the conversation over to Stephanie, who will ask you some background information.

DEMOGRAPHIC INFORMATION

- What is your age?

- What field(s) are you trained in, what is your occupation?

- How long have you been working in your field?

BACKGROUND INFORMATION

- What kinds of subtyping methods are you most familiar with? What do you like or dislike about these methods?

- Do you work with any infectious agents? Which ones?

- In what research context do you usually work with MLSTs and/or other subtyping methods? (e.g. with my collaborators in _____ field of research, it's part of my personal research...)

SCENARIOS

Open scenario: Could you walk us through a research scenario(s) in which you have worked with MLSTs?

Follow-up Questions

- What challenges did you encounter?
- What resources did you turn to for help?
- Were you able to solve the problem with available resources?

Appendix C: Formative Usability Evaluation Protocol

INTRODUCTION

Thank you for choosing to participate in the second phase of our research for a next-generation MLST tool to be incorporated into our PATRIC bioinformatic website here at VBI. I am Stephanie Alpert, and I have been doing R&D for PATRIC as part of my master's thesis. I'm here with Joe Gabbard, a research faculty with expertise in human-computer interaction and user experience, and Meredith Wilson, a Biosciences Analyst with a background in epidemiology.

Today we're going to talk and walk through some tasks with the MLST tool for about 90 minutes. We will be working with a PATRIC prototype that allows you to browse the preexisting PATRIC MLST sequence data, or bring a single sample of your own data to compare to PATRIC data.

You will be working through several different screens, and at various times we'll be asking you questions that require to you locate and parse various pieces of information contained on the screen. Some questions will be strung together in a task flow, others will stand alone and require you to hop from one area of the screen to another. All questions we ask will have answers a that are "findable" somewhere on the screen or at least accessible through a widget on the screen.

For the purpose of this study, we ask that you only interact with the prototype when prompted by a task. There will be time at the end for free exploration. Additionally, please think aloud as you are working through the tasks so we can better understand your thought process as you interact with the prototype.

Do you have any questions for me at this point?

Please keep in mind this is not a test and it does not matter how well you can complete the tasks. Just being able to observe your interactions with the prototype will help us gain valuable information for its improvement.

TASKS

1. You are investigating a group of patients from a local hospital who are presenting with gastroenteritis and are all seropositive for E. coli. All of the patients are from the same University. You receive individual PCR sequencing of MLST loci extracted from the patient samples, and the sequence data for the individual genes.

You also get samples and corresponding MLSTs from suspected sources in the university dining hall.

Is this clear? Do you have any questions so far?

For the purpose of this task, we will only have you investigate one patient and one sample (beef). Additionally, please use the upload mechanism for this task. Did the beef sample cause the patient to become ill?

a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?

RESULTS PAGE - Describe what you see on this page? What are your first impressions?

LANDING PAGE - Now please click on ST335 in the table. Can you please describe what you see on this page?

2. Which sequence type has the most E. coli genomes that contain the "icd 12" gene?
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

3. What can you tell us about the evolutionary history of ST335?
    a. What made you use eBURST or the phylogeny?
    b. Is there anything missing that would help in your assessment?
    c. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?

4. What are the clonal complexes for ST335?
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

5. Which genomes are most closely related to E. coli DEC5A?
    a. What do you think of the color key?
    b. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?

c. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

6. What is the most closely related ST to ST335 in this phylogeny?
    a. What other information besides sequence type might be useful to add?
    b. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    c. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

7. Which E. coli genome on record was collected the earliest?
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

8. Can you display a geographical representation of ST335 isolation sites?
    a. Is there any additional functionality you'd like to have with the map?
    b. What would you expect to happen if you click on a map pin?
    c. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    d. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

9. Have there been any recent publications on E. coli ST335?
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

We are now going to explore another sequence. Don't worry about any sort of backstory. You know you have an E. coli sample, but you don't know much else. The general goal here is to explore the interfaces, and gain knowledge/insight about the sequence.

10. Here are the sequence data for a particular sample. Please use the MLST tool to explore this data by pasting the data in FASTA format on the first screen.
    a. Tell us what you see in this result set? What stands out? How do you interpret this data? What does it mean to you? What is confusing?
    b. What is missing that would help make more sense of this data?

c. Does this genome match any known ST exactly? Are there any close matches?
d. What is the closest match to this sample?
e. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
f. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

11. Characterize the differences between your data and ST11.
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

12. What are some closely related alleles to the fumC allele from your data?
    a. Is there any other information that would be useful to present for each of these alleles?
    b. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    c. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

13. Is what is being labeled currently as your unknown fumC, actually a fumC allele?
    a. What on this page is useful/not useful?
    b. What other options would you want from BLAST?
    c. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    d. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

14. You want to get the sequence for icd 54. Please generate the data for this purpose.
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

15. We know that fumC 73 is about 90% similar to yours. Where do the differences lie between them?
    a. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?

b. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

16. Is the difference between your fumC and fumC 73 typical of the locus?
    a. Is there any other information you'd be interested in seeing here?
    b. How easy was this task on a scale of 1 to 7, with 1 being very hard and 7 being very easy?
    c. How useful do you find this capability on a scale of 1 to 7, with 1 being not useful and 7 being very useful?

POST-TEST QUESTIONNAIRE

- Is there anything else you would be interested to explore?

- What else would you be interesting in doing from this page (comparative landing page)?

- What do you think the most useful feature of the MLST tool is? The least useful?

- Is there anything you would change?