Case-Based Stories for Traceability Education and Training

Orlena C. Z. Gotel
Independent Researcher
New York City, USA
olly@gotel.net

Stephen J. Morris
Department of Computing
City University London, UK
sjm@.city.ac.uk

Abstract—**The purpose of this paper is to disseminate knowledge about proven techniques relevant to software and systems engineering, and in particular to requirements traceability, but derived from another industry and another professional approach. It describes the process of investigating and dealing with an outbreak of a foodborne disease, an established process that depends upon traceability practice at multiple levels and its alignment within two distinct disciplines. We tell a story based upon an outbreak investigation case study to step through the details of this process and to discuss parallels for the practice of traceability in the context of identifying and addressing requirements-borne problems. Not only is this process a model of industrial practice in these other two disciplines, the provision of case study materials based upon past events is also a model industrial practice that makes the details of the process widely accessible for education and training. This paper seeks to encourage the similar use of case studies in requirements engineering education and training, to develop realistically grounded stories that can be used to step through the investigation of a requirements-borne problem, and so to demonstrate and discuss the practice and value of requirements traceability at each step.**

Keywords-case studies; epidemiology; food traceability; requirements-borne problem; requirements traceability; stories; tracery; tracing; tracking.

I. Introduction

One of the challenges for software and systems traceability is in making traceability itself fit for purpose [7]. While it is common to state that traceability is required to support requirements change impact analysis, or the validation and verification of requirements, it is rare to examine these tasks in detail to elicit the nature of the traceability they demand. One of the other challenges facing traceability is demonstrating its value to those who would fund or implement it [7]. It is imperative to share more industrial experiences and data to address these challenges.

In a previous paper [8], we examined the concepts underlying tracking and tracing in a number of disciplines (animal tracking, art provenance, epidemiology, food traceability, luggage handing and metrology) and emphasized the importance of proactive sign making as a core element of good practice, guided by the principle that "there is no ability to trace without a track and there is no ability to lay a track without making signs". We also argued elsewhere [9] for the importance of "tracery", defined pragmatically as "a pattern of traces" or more generally within software and systems engineering as "a web of interconnections created alongside and within all the important development artifacts". In this paper, we examine two of these other disciplines in more detail, to show the role of the tracery within them and to disseminate knowledge about their proven traceability practices. The demonstrable success of these practices reveal that they are both fit for purpose and valuable, suggesting that these disciplines may provide a model for requirements engineering to emulate.

Their examination also provides a means to stimulate discussion about a number of topics relevant to requirements engineering education and training, including: the nature of traceability; desirable levels of traceability detail; recognition of failure symptoms and their diagnosis; early warning systems; necessary remedial actions, processes and co-ordination; pre-emptive action via original development processes; and training for all aspects of desired processes.

Epidemiology is the study of the patterns of diseases and their causes [4]. It establishes the usual distribution of a disease in terms of symptoms and causes, population effected, duration, location, season and other factors. An outbreak occurs when the level of a disease exceeds that normally expected. Much epidemiological work therefore depends upon the collection and analysis of large data sets. Food traceability provides the ability to trace back to the origin of any component that has been integrated into a food product [13]. It provides the ability to examine both the chain of production and the chain of distribution, thus supporting analysis and recall activities.

Software and systems engineering lacks the formal working procedures specifically intended for tracing that are now widely used in the successful investigation of epidemics and their termination. These facilitate the rapid tracking of potential patients, the identification of illness and the isolation of causes. The main effort is essentially required after the event, looking backward at what has happened recently. The food industry has put similar efforts into product tracing procedures, the principal difference being that the effort must be expended before the event, looking forward to the possible need to make a food recall. Software and systems engineering also lacks the detailed case studies, evident in both disciplines, for the dissemination of tracing practices and for conveying their role within a complex process. However, formal working procedures are not alien to software and systems engineering, the most established that are related to traceability being those for debugging and security. These procedures are reviewed in Section XI.

The paper is written around a story involving disease tracking and food tracing. Its significance lies in the synthesis that it provides of the many detailed case studies of tracking and tracing which continue to be published in the epidemiology and food traceability literature. While the story is fictional, it is based upon the training materials provided by the Centers for Disease Control and Prevention (CDC) and, in particular, "A Multistate Outbreak of *E. coli* O157:H7 Infection" case study [19]. Such materials step through the stages of an investigation, revealing information progressively and punctuating the narrative with questions suitable for classroom discussion or for guiding personal study. The story of this paper therefore provides a realistic example of how a foodborne disease outbreak would be investigated and dealt with, and clarifies the tracery it sets up and the traceability it employs. It sets the scene for thinking about a mature process in software and systems engineering. Stories from other domains have proved useful for learning about software design [6] and a similar storytelling approach may be a viable strategy for demonstrating the value of requirements traceability in a practical context. We use the outbreak story to seed the development of an analogous trace story in software and systems engineering, the creation of which could be undertaken as a classroom exercise and based upon experiences. Just as the other disciplines use such material to stimulate structured discussion, we further highlight the parallel questions this approach could raise.

II. Back Story

Your presentation at the International Requirements Engineering Conference was a great success this year, but your flight home is interminable! You feel nauseous and your stomach is cramping. You have had to get up to use the lavatory multiple times and you do not like what you see. You begin to sweat and feel that you have the onset of a low-grade fever. By the time you reach home, you are convinced that this is not your usual bout of traveler's food poisoning. Twenty-four hours later and your stomach pains are severe. You are worried about all that blood you see and you are finding it difficult to quench your thirst. But, as there are still no alerts from your colleagues, you conclude that it must just be you. Researcher that you are, you make your own diagnosis from the web and rush to the doctor's surgery.

Back in the conference city, clinicians have just reported a 300% increase in laboratory reports of *E. coli* infection over the past month, as compared with the same month in previous years, with the bulk of the cases appearing over the last few days. The Dept. of Community Health immediately call their newly-appointed epidemiologist: "Dr. Epi, is this a foodborne disease outbreak? What should we do?"

Dr. Epi is a recent graduate and this is the first time that she has been posed this question when she holds responsibility. She knows that there are standard ways to conduct an outbreak investigation, so she uses the step-by-step guide of the CDC [3] as her plan, augmented by material from the World Health Organization [20] and numerous course materials [1,2,15,18]. The basic steps of Dr. Epi's investigation plan are shown in Figure 1 and elaborated in the following sections. With the mention of *E. coli*, Dr. Epi knows that she will have to undertake both an epidemiological and environmental (food) investigation in parallel, with the findings of the epidemiological investigation guiding the collection and testing of the environmental samples. The date is October 1st, 2012.

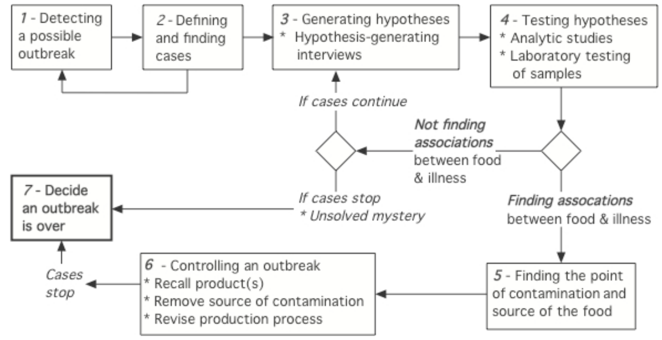

Figure 1. Steps in a foodborne outbreak investigation (adapted from [3])

A. Analogy and Discussion Questions

There are numerous deployments of a particular software system around the globe that serve to facilitate the work of air traffic controllers. Because the system concerned integrates software that has been developed in different places, using separate processes and teams, and satisfying separately defined requirements, there are many junctures at which a problem (c.f., a pathogen) could have been introduced. Just as distinct vectors carry specific diseases, there can be different reasons as to why problems materialize in software systems (e.g., requirements-borne, technology-borne, user-borne, environment-borne, etc.) Across these particular deployments there are the occasional error messages and anomalous behaviors in end-use that all fall well within normal operating parameters. However, they have begun to increase in frequency and mutate in form, and only one operator in one location has taken note. Controller Charlie ponders whom he should tell and what he should do.

In those settings where we make use of software systems, do we have a monitoring and early warning system for problems as we do for disease, a way to recognize atypical situations at both a local or global level, and an authority to notify? On what class of software system would it be necessary to set up a monitoring and warning system like this, and how could it be adapted to the criticality of the application domain or the speed demanded for action? Of what would such a system comprise, and how would it be set up and used? What real-time data would need to be gathered to accumulate trend data? Whose role would it be to orchestrate and coordinate this process in an organization? What should Charlie's responsibility be in all this? Moreover, how can we recognize when the symptoms of a problem are requirements-borne? Are there any guiding professional agencies or go-to step-by-step conceptual processes to direct us in how to investigate and act on such problems? If yes, how do they operate and how useful are they? If no, should there be, how could they operate and what barriers would be presented? Affirmative answers to these questions would suggest that the foundations of a tracery are already in place for the discipline of software and systems engineering.

III. Detecting a Possible Outbreak

The Dept. of Community Health in the conference city contacted Dr. Epi. They have a public health surveillance system in place in their laboratories and an astute clinician had recognized an unusually large cluster of *E. coli* infections. The first thing that Dr. Epi did was to re-familiarize herself with the disease under investigation. She then reviewed the baseline data to verify the diagnosis of *E. coli*. Any potential for diagnostic error had to be ruled out before launching a full and costly investigation. This was achieved by verifying the signs and clinical symptoms of a few of the patients. Dr. Epi found that the symptoms were reported accurately and were compatible with the diagnosis of *E. coli*. This was coupled by verification that the correct standardized tests for detecting *E. coli* infection had been undertaken appropriately and reported correctly.

Dr. Epi then needed to confirm the existence of an outbreak. An outbreak is defined as: "a situation when the observed number of cases exceeds the expected number of cases of a specific disease in a given population for a given period of time" [15]. Was this a true outbreak with a common cause or simply unrelated cases of the same disease? The same illness was being reported in a common place and DNA "fingerprinting" of the bacterium indicated the same parent. Using local health department records, Dr. Epi was able to assess the normal background activity of the disease and confirm that the recent number of cases was unusually high for the location and time of year.

The all-important questions then became: How urgent is an investigation? Is exposure to the source of the disease still happening and, if so, who is the population at risk? To answer these questions, Dr. Epi needed to collect much more data, locate the source of the infection and prevent additional cases from happening. The outbreak appeared to be local at present, but Dr. Epi knew that it could take a couple of weeks for full data to become available.

A. Analogy and Discussion Questions

The challenge in software and systems engineering also becomes characterizing the symptoms of a problem, determining whether occurrences are all due to the same underlying cause, assessing the likely impact if a problem is left unchecked (i.e., its path and scope), agreeing the urgency with which a problem needs to be addressed and discovering what is causing the problem in the first place (i.e., where did it get introduced and how is it being disseminated). As with disease, any investigation can be compounded by the fact that there may be an incubation period before the symptoms of a problem become visible; requirements-borne problems may lie dormant for a long time. Controller Charlie contacted the lead contractor for the air traffic control software system, who duly consolidated all the diagnostic data from all the various deployments and sought to look for patterns.

In software and systems engineering, do we routinely collect historical data and use statistical knowledge of "background activity" to inform analyses, both within and across deployments or projects? What historical data would have been useful to collect in the case of the air traffic control software system to aid diagnosis? Could you see value in having a repository of commonly known requirements-borne problems that have well-defined characteristics that can be tested for, showing typical patterns of behavior and scope of impact? What checks and balances could we routinely employ prior to investing time and effort in any investigation? How do we make an agreed diagnosis of the problem we are investigating? How well do we really need to understand the "intangible" things that we may attempt to subsequently trace? Tracery construction, as this story shows, may exploit existing organizational structures and data sources as a foundation for making special checks and facilitating analysis at an early stage in the investigation.

IV. Defining and Finding Cases

In conjunction with her initial investigations, Dr. Epi began to assemble a multidisciplinary outbreak investigation team. Their expertise would span clinical medicine, epidemiology, environmental and public health, and food microbiology. Once assembled, the team began to put together a case definition to identify those cases that could be associated with the outbreak. A case definition provides a simple and uniform way to identify patients as cases of the disease, comprising inclusion and exclusion criteria: clinical (symptoms, signs and onset); epidemiological (person, place and time); and laboratory (culture results and dates).

For the outbreak investigation, a case was defined as diarrhea and/or abdominal cramps, in a resident of the conference city or its surrounding suburbs, with the onset of symptoms from September 24th and a stool culture yielding *E. coli* with the outbreak strain DNA "fingerprint". The case definition would help to determine how many cases there actually were by pinpointing when, where and who. However, it would not include patients who did not see a doctor and it would exclude visitors to the conference city.

Dr. Epi further requested that the cases be classified as: confirmed (clinical symptoms characteristic of the disease, plus a positive test); probable (symptoms clinically confirmed to match disease, but no test or epidemiological link established); or suspected (symptoms reported to match the disease, but without confirmation). Within the first twenty-four hours, the investigation was centered on the conference city and its suburbs. Medical records were examined and the team conducted standardized interviews of all known cases, irrespective of classification.

The team found that there were three confirmed cases over the statistical norm on September 27th, three over on the 28th, seven over on the 29th and twenty-one over on the 30th. Today was still October 1st and Dr. Epi was anxious as the epidemic curve (a bar graph that plots the progression of cases of the outbreak over time) confirmed that the outbreak was ongoing. *E. coli* has a typical incubation period of three to eight days, so while the team could establish the onset of the outbreak from between 20th to 25th September, they could not yet know its duration, so predict its scope. The team requested that health care providers, emergency rooms and laboratory professionals report any related illnesses quickly.

The team organized the preliminary interview data they had collected for analysis. When analyzing common factors, over 50% of the cases were found to be men in the twenty-one to forty-nine age ranges. The place data plotted on a spot map showed a spread of cases around the conference city, with a hub near a neighborhood known for its restaurants and bars. While the team was concerned that the source might be one of these, Dr. Epi needed well-defined hypotheses to help her team gather more data and act responsibly. Since the cause was still uncertain, effective control measures could not yet be implemented. However, all exposed individuals were to be treated with antibiotics and monitored.

A. Analogy and Discussion Questions

The tracery in the above story is now specific to the type of disease in hand and spreading out along paths that are already known. There is extensive data collection of predefined types and the result is a refined picture of the situation generated by an established process. In software and systems engineering, any investigation of a requirements-borne problem would similarly benefit from the expertise of various stakeholders. The lead contractor assembled a team from the development and deployment sites and started collecting and analyzing more data.

While a multidisciplinary team may be responsible for creating the deployed software system originally, what kind of team should be assembled for identifying and addressing a requirements-borne problem within it? What skills are required? What data would need to be collected to advance the investigation and how could these be obtained? What tools could be used to examine these data from multiple perspectives and support analyses, like epidemic curves and spot maps, and in what way could statistics be exploited to determine patterns and trends? How could we characterize "cases" to identify, with certainty, those errors or anomalous behaviors that are caused by the same requirements-borne problem? It is common to deploy temporary fixes for software systems (i.e., patches), but how are they monitored for effectiveness and analyzed for impact elsewhere?

V. Generating Hypotheses About Likely Sources

The team examined the records of previous *E. coli* outbreaks to gain more clues about the likely source. To support hypothesis generation, the team collected data from patients on where and what they had consumed over the week prior to the first samples being taken, and the team asked each case to recall their meals one by one. They focused upon the consumption of exposures linked to *E. coli* infection in the past. An infection arising from a vegetable source has tended to have a demographic different to the one being found, the most common explanations for young to middle aged men being contaminated meat and beer.

The team found one common element in nine cases, a visit to a British pub and a meal of bangers and mash, with beer. A potentially linked case was an elderly couple who reported a breakfast of sausages, beans and eggs a few days before they felt unwell. Another comprised eight people out of ten who had recently had a BBQ and consumed burgers and hotdogs. Yet another case involved a church supper, where all twelve people eating the meatloaf became ill. Each exposure suggested a separate hypothesis, but the team was converging on a meat or beef source as the main possibility.

Suspecting a pub meal as one possible source of the contamination, the Dept. of Community Health contacted the pub owners, and bangers and mash was withdrawn from the menu pending further investigation. The various home-brew beers were also suspected, but there were not sufficient data to justify their withdrawal. With the breakfast, BBQ and church supper cases, the meat sources and its preparation became the focus of additional data gathering.

A. Analogy and Discussion Questions

In the above story, the pattern of the tracery is now becoming more complex and spreading out in different directions, but the structure enables decisions to be made and preliminary action to be taken, blocking off particular paths. What is pertinent is how extensive and labor intensive the data gathering is, and the role of educated guesses to form hypotheses and scope this work. The process continues to be supported by historical data and dependent upon the expertise of those involved. Detecting the source of a requirements-borne problem in the air traffic control story relies on a similar tracery. The lead contractor has a requirements traceability matrix in place that links requirements through design to code, but there are gaps in the subcontracted work following early versioning. The lead contractor sends an analyst to interview the subcontracting team in an attempt to re-establish this traceability.

In the analogous story, would time be taken to form multiple hypotheses or would the problem solving orientation of our discipline lead to premature convergence on a source? Do we collect sufficient data to reconstruct activities that led to the emergence of the problem (i.e., the actual use of the software system over time by numerous parties in different contexts)? How can we account for evidence that is absent or has vanished? What could have been done earlier to avoid a paucity of record? If the investigation leads to a function in a software component that is capable of generating new cases of the problem, what could be done to ensure that its "withdrawal" from the overall system is unproblematic, and how?

VI. Testing the Hypotheses

The sausage stock at the pub proved negative for *E. coli*. The pub did report, however, that they receive a regular delivery of British-style sausages, so any rogue batch causing infection had already been consumed. The elderly couple did not have any sausages left over from their packet to test, reporting that they receive a packet of farm-fresh sausages weekly from their son and tend to eat them all the next day. The BBQ family reported that they bought their sausages from a named supermarket but they made their own burgers from some ground beef that they bought. The churchwarden reported being given a donation of ground beef for their supper from one of the parishioners. The team inspected all the premises of food preparation and they all proved negative for *E. coli* contamination. The BBQ had been dismantled, but the team did notice that the church oven was operating at a lower than specified temperature, so the team began to suspect that the meat had been undercooked.

The team discovered that the pub sausages were made locally with the ingredients sourced from local farms. They also discovered that the son of the elderly couple visits a small farm west of the conference city most weeks to buy sausages direct from its shop, the same shop that the BBQ family visited to buy their ground beef. The ground beef used in the church supper was further found to come from a farmer's market affiliated with the farm shop.

To develop and further test the evolving hypothesis on the source of the outbreak, the team conducted a case-control study. Two controls were selected for every patient case and matched by age and gender. These were persons without the disease but representative of the population from which the cases originated. In the case-control study, 100% of the ill persons reported eating ground beef in some form or other in the week before the onset of the illness. 9% of the control persons reported eating sausages, 15% reported eating burgers and 18% thought that they had eaten ground beef at some point. No other food or beverage was significantly associated with the illness. Since ill people reported eating ground beef more often than well people, it was associated with the illness and the team undertook further statistical tests to determine the strength of the association.

A. *Analogy and Discussion Questions*

At this stage in the above story, the tracery links together a number of analytical studies testing primary and secondary hypotheses. These tests, samples, inspections and interviews provide data that are intended to assess the relationship between a given exposure and the disease, and to determine the statistical significance of any such relationship. The studies form standard and recurring elements within the tracery structure, their exact position being determined each time the process is run and the whole tracery activated. When software components are sourced in different places, integrated in other locations, distributed and sold by various re-sellers, and then used in diverse contexts, it makes it challenging to get to the bottom of a problem that may be due to the same requirement.

In the analogous story, what kind of plausible hypotheses could we form and test? We routinely investigate the software product, but what added value is there to investigating the various development processes employed and their wider environments? What would be the equivalent of a case control study in the air traffic control story? How could statistics be used to confirm the correlation between a problem and a candidate requirements source? Do we also investigate compounding evidence or treat it as noise?

VII. Finding the Point of Contamination (Source)

Although Dr. Epi had solid epidemiologic evidence to implicate ground beef, additional studies were still necessary to locate the actual source of the bacterium. Because the outbreak was linked to meat prepared in multiple kitchens, contamination is likely to have happened somewhere earlier in the food chain. To investigate this, Dr. Epi co-opted two regulatory compliance inspectors and requested a traceback of the sausages to the distributor, processor and producer, and an examination of the entire chain of production to see whether the beef within it was linked to the beef in the other cases. If it was the same beef, where else had it been sold and what other food products had it been used in? Was it causing cases elsewhere? The scope was as yet unknown.

A traceback determines the production and distribution chain of a food product. It is intended to identify sources of contamination and remove them, affecting a food recall if necessary. It begins with the information available at the time of purchase of the implicated food item and extends back to the beginning of its production, examining all intermediary steps. A full regulatory traceback follows rules of legal evidence and may require hours of work if records are poorly maintained.

The traceback began at the pub where the sausages were purchased, cooked and served. There was no leftover packaging from the sausages used prior to the onset of the outbreak but, as the pub used the same supplier every week, the team was able to locate the distributor. According to the purchasing records of the distributor, who retained all traceability information from product labels electronically (e.g., product name, size and weight of package, code and lot numbers, manufacturer name and address, dates shipped, received and purchased, etc. [19]), the sausages in question had come from a family-run sausage processing facility in the region. The team found that this facility had passed a sanitary inspection recently, and its sausage making process and equipment was exemplary. The family also maintained meticulous records of their workflow and the sourcing of their ingredients. The "one step forward, one step back" traceability principle had been followed, and both external and internal traceability had been established (see Figure 2) [13]. The staff was trained to the highest food handling standards and none had been ill over the past month.

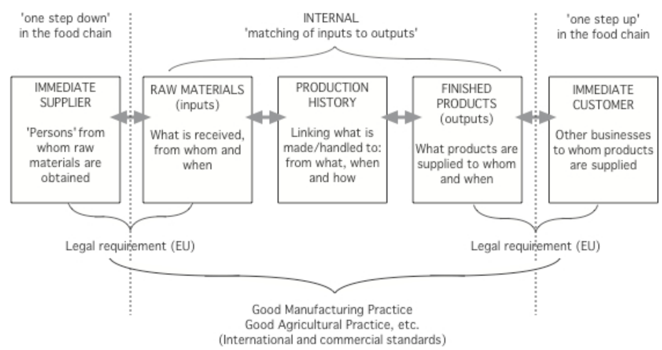

Figure 2. Basic components of traceability in a food chain [13]

Concurrently, the team visited the farm shop to instigate a traceback on its packaged sausages and ground beef. This led to a different sausage processing facility. Because the overall traceback led to two sausage-processing facilities, it appeared that the contamination occurred before the processing. The point of the contamination was likely to be in a beef supply common to both, so the investigation sought the point at which the pathogen could have been introduced.

The ground beef used in both sausage-processing facilities was recorded to have come from a particular meat-grinding facility, one that both had used while their usual facility was closed for an equipment upgrade. This facility was inspected, along with the records of the source of all the beef that was ground during the period. While undergoing this sanitation inspection, the slaughterhouse supplying the beef was identified and also inspected, as were the farms that sourced the cattle. Each step in the processing, storage and transport of the beef supplied to the facility was meticulously recorded and now examined. The farms sourcing the cattle had used industry best practices for cattle identification and had complete histories of the life and feed of their cows [10]. All the farmers worked with a slaughterhouse that retained traceability for every beef product they received, cut and packaged. Attention thus turned to the meat grinding facility.

The facility under inquiry was found to be over-due an inspection. Swabs were taken from everything that may have come in contact with the suspect food. The usual workflow was also examined and it was found that the grinders were rinsed routinely with water that was stored in an outdoor container in between disinfections. The team spotted deer on the facility and considered whether the water supply could have become contaminated with fecal matter from deer; deer are known to be a reservoir for this particular strain of *E. coli*. This water tested positive for the pathogen.

A. Analogy and Discussion Questions

There are preparations that have to be put in place at every stage of the food chain to enable a transparent traceback and recall, and some sectors of the food industry (e.g., the beef industry) have exemplary practices and recall preparedness [10]. These include the clear identification of a traceable item, the use of unambiguous labeling and industry codes, and the collection and exchange of industry-agreed data. Furthermore, there is a strong concept of custodianship, with care taken to transfer this responsibility at all points of transformation. However, even when an industry is properly prepared, doing a traceback is never trivial; it incurs considerable effort and relies upon the collaboration of all traceability partners in the chain.

In this second stage of the investigation, the basic structure of a new tracery is predetermined by the "one step forward, one step back" principle and the detailed records of inputs, outputs and intervening processing required in the food industry, and legally enforced in many countries. This pre-existing regulatory structure transforms the tracking activity from a necessarily *ad hoc* albeit highly organized process into an activity which essentially involves following a carefully laid trail of evidence from the final destination of any food component to its exact place and time of origin.

In software and systems engineering, we can be reluctant to put traceability in place as a pre-emptive measure, unsure of the cost-benefit, and may attempt to recreate a traceback in time of need. In the analogous story, the first link in the chain is as critical as the beef farmer, and this puts requirements engineers in a key position. It is the preparations that these stakeholders put in place that establish the potential to trace and set the precedent as to how. What could they have done to prepare for this investigation? What is the role and responsibilities of the other stakeholders in the development chain? In this particular story, the recreated backward traceability led to a design element in a component of the overall software system that had been designed and developed in Europe. The subcontractor had interpreted the measure expressed in a requirement using metric units as opposed to the intended imperial units of the US-based lead contractor.

What agreement is there upon those data that need to be collected, recorded and shared about traceable items in different software domains (i.e., its signs)? What shared policies and mechanisms exist for defining and handling artifact transformations, be they internal or external, within or across organizations (i.e., its tracks)? When attempting to recreate a traceback later, and where automated techniques are unavailable, how would we know we have located the correct starting point? What step-by-step guidance is there for undertaking this tracing retrospectively and manually? When would regulatory agencies get involved?

VIII. Controlling the Outbreak

With the source of the outbreak pinpointed, the team needed to design, implement and evaluate control measures to prevent further illness and future outbreaks. First, the meat grinding facility was closed temporarily to address the contaminated water supply. Second, while the quantity of ground beef was not large and no further products were found to use this particular supply, it was evident that there should have been more cases than the team saw from the records. Consumers were therefore informed to throw away any suspect meat they had frozen, and details on the lot of the meat was available from the traceback to assist this. Third, a more general notice was disseminated to the public to remind them to cook all meat thoroughly to kill any pathogens. Monitoring was put in place by Dr. Epi to ensure that the control measures were executed and working.

A. Analogy and Discussion Questions

At this stage the tracery structures have served their purpose, but questions remain outstanding. How easy would it be to affect a requirements recall in the analogous story? Do we ever simply remove a requirement and all the components it touches, or do we alter it in some way and then attempt to remedy the knock-on problems in all affected components? What tools do we have to gain transparency into our efforts and assess their effectiveness? Do software consumers have an equally vital role to play in the process?

IX. Deciding the Outbreak is Over

By October 10^{th}, the number of new cases of *E. coli* had returned to background levels. Dr. Epi determined that the outbreak was over and that the source of the infection had been eliminated. The team put together a summary of the investigation and its recommendations, and disseminated them to team members, the Dept of Community Health, and the epidemiology and food industry communities, to share case study knowledge and prevent future similar outbreaks.

A. Analogy and Discussion Questions

Traceries are not fixed structures and demand review and maintenance. Requirements from the analogous story, and their satisfaction in design and code, may be used in other software systems, so any problem identified and addressed may re-emerge. Do we continue to gather data to trace and measure once a problem is "closed"? What would be an effective way to share experiences to advance understanding and provide resources to improve collaborative practice?

X. Story Epilogue

Your own doctor provided a rapid and accurate diagnosis, your prescribed course of antibiotics worked and you made a full recovery. However, no one ever discovered the source of your *E. coli* infection. No one thought to make the link between the solitary cases outside the conference city area. This is because elsewhere the infections did not register. Moreover, you never even realized that there had been an outbreak in the conference city and did not check.

You consider yourself a pioneer of traceability research in the requirements engineering community. As you read this story, you are impressed at the investigation and by the enabling role of its traceries, and you can see that this provides an excellent case study for others to learn from in these disciplines. You find additional case studies that both alarm and comfort you. You wonder: Where are the step-by-step guides that explain how to undertake the investigation of a requirements-borne "disease" in software and systems engineering? Where are the training materials that describe how to establish and make use of traceability in practice to support this? You know that your discipline is uniquely complex, but you can see that the work involved in these other disciplines is not trivial. As you plan your conference submission for 2013, you vow to find a way to encourage the dissemination of traceability case studies to create similarly detailed stories to learn from, and to also transfer any immediately applicable practices into your own discipline.

XI. Related Procedures

Formal working procedures are not alien to software and systems engineering, the most established being those used for debugging, which are similar to the process of Figure 1. First, any failure has to be replicated (Steps 1,2), and then the root cause has to be identified through hypothesis testing (Steps 3,4). The defect has to be fixed and the fix needs to be distributed (Steps 5,6). When projects must deal with security vulnerabilities in their software, there are strict guidelines and processes to follow, a situation similar to epidemiology with the vulnerability being the disease.

The field of software and system security is one where standards, procedures and processes have been developed to a high level, motivated by the general importance of the issue and by specific requirements, many backed by law, for the security of financial transactions and other critical systems. Two international standards, BS ISO/IEC 27002:2005 [11] and BS ISO/IEC 27001:2005 [12], set an overall framework. This section provides examples of important aspects of the activities covered by this work.

The Open Web Application Security Project includes among recommended objectives the consideration of security explicitly during the software requirements process [16]. This involves, inter alia, the derivation of security requirements from business functionality, the conduct of a review of functional requirements that specify the business logic and overall behavior for each software project and, after gathering requirements for a project, the conduct of an assessment to derive relevant security requirements. For each functional requirement, a security auditor should lead stakeholders through the process of explicitly noting any expectations with regard to security. Typically, questions to clarify for each requirement include expectations for data security, access control, transaction integrity, criticality of business function, separation of duties, uptime, etc.

Major manufacturers all provide more or less detailed guidance about the procedures that should be followed for manual or automatic searches for computer viruses and for actions to be taken. The guidance from Dell [5] says that when a virus is detected it is necessary to immediately locate and scan the relevant file(s) with anti-virus software to determine if the virus has been immunized, to communicate a virus alert to warn staff of the incident and the appropriate response, to establish whether the virus might have infected others and, if so, to respond accordingly. Following the virus attack, it is necessary to review the measures taken to minimize damage and prevent a recurrence, and to question whether procedures and safeguards remain adequate.

There are also major resources for checking the nature of viruses and other vulnerabilities. For example, the U.S. National Institute of Standards and Technology maintains the National Vulnerability Database [14], a searchable index of information on computer vulnerabilities. As of July 2005, this Metabase contained 10,619 vulnerabilities, exposing organizations to risk of attack. Analysis of these data indicates that software accounts for more than 94% of the vulnerabilities.

Detailed guidance is also available for procedures to be followed when a purchaser of software wishes to alert its vendor. For example, Figure 3 illustrates part of the advice give by the Organization for Internet Safety (OIS). It shows the steps in the investigation process to be carried out by a vendor on receipt of a Vulnerability Summary Report (VSR) from its Finder. According to this procedure the Finder should expect to receive a reply to a Request For Status (RFS) within a prescribed period.

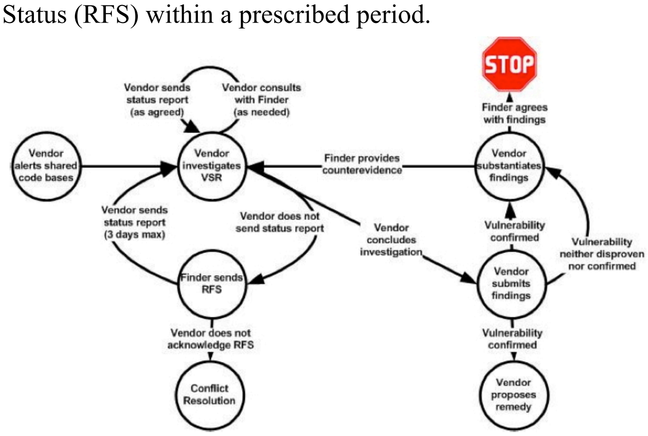

Figure 3. Steps in the investigation phase (OIS guidelines) [17]

This field is noteworthy for the emphasis placed on an immediate focus on rapid tracing, diagnosis and treatment, combined with a much wider view incorporating all aspects of development and management. The potential commercial impact of side effects such as loss of reputation, what in the medical world might be called "secondary infections", shifts the emphasis. Tracing a virus of any kind back to its original source is a lower priority, at least for most commercial organizations, than the rapid application of a cure.

The ideal tracery structure appropriate in the security field appears to be more than local, built up within an organization by means of procedures that extend from the earliest stages of development through to post virus eradication reviews and, at the same time, extending outwards to directly linked organizations that share a common interest in security, most importantly the customers or suppliers of software systems. The potentially damaging effects of virus infection, both primary and secondary, provide the obvious motivation for all the work in this field, in contrast to that in the general field of traceability studies.

XII. Conclusion

There are two distinct tracing components in this foodborne disease outbreak investigation. The first component is a tracery structure [9] on a local and national scale, a linked set of agencies, responsibilities and analysis techniques that can be brought into operation whenever needed to identify and track all aspects of an outbreak. This complex tracery of people, roles, expertise and authority essentially lies unused and dormant until needed. The second component is a permanently running system that records data and builds up another tracery structure, which provides immediately available trails [8], on demand, for all food products and their components. This second tracery requires maintenance by the entire food industry, ultimately enforced by government regulation.

The first tracery enables investigations to "fan out" and cover the wide range of issues related to disease outbreak; the second then enables investigations to "fan in" and identify the origin. This combination of interlocking traceries provides a model for practice that might be followed in software and systems engineering. An alternative tracery model is also needed in order to take account of the varying levels of incentive, and therefore of motivation, that are immediately apparent when considering the variety of software systems within which traceability might be an issue.

Traceability is a mechanism that supports and enables many important tasks in software and systems engineering [7]. By developing case-based stories, grounded in actual industrial experiences, we would place requirements traceability in context. We suggest that this is essential for conveying its practice and value in educational and training settings. By examining our discipline's stories alongside those of other disciplines, we would further highlight how other professionals routinely collaborate to identify and address problems that demand traceability. This would offer students the potential to assess, learn from and maybe even leverage some of the many distinguished practices therein.

Acknowledgements

We thank the reviewers for their constructive feedback.

References

[1] L. Alexander, *et al.*, "An Overview of Outbreak Investigations," FOCUS on Field Epidemiology, vol. 1(1), North Carolina Center for Public Health Preparedness, University of North Carolina at Chapel Hill, 2003.

[2] Center for Infectious Disease Preparedness, "Conducting an Outbreak Investigation in 7 Steps (or less)," University of California Berkeley School of Public Health, June 2006.

[3] Centers for Disease Control and Prevention, "Step-by-Step Guide to Investigating Foodborne Outbreaks," CDC Atlanta, GA, *www.cdc.gov/outbreaknet/investigations* (Jan 2012).

[4] D. Coggon, G. Rose, and D.J.P. Barker, Epidemiology for the Uninitiated, 4th ed. British Medical Journal Pub. Group, 1997.

[5] Dell Technical Support. Virus/Spyware Troubleshooting Guidelines, *support.dell.com/support/topics/global.aspx/support/security/security_troubleshooting?c=us&l=en&s=gen&~mode=popup* (@May2012).

[6] S. Fickas, "Desert Island Column," Automated Software Engineering, vol. 1(1), 1994, pp.141-146.

[7] O. Gotel, *et al.*, "The Grand Challenge of Traceability (v1.0)," in J. Cleland-Huang, *et al.* (eds.), Software and Systems Traceability, Springer-Verlag, 2012, pp.343-409.

[8] O.C.Z. Gotel and S.J. Morris, "Out of the Labyrinth: Leveraging Other Disciplines for Requirements Traceability," Proc. IEEE Intl. RE Conf. 2011, Trento, Italy, pp.121-130.

[9] O. Gotel and S. Morris, "Requirements Tracery," IEEE Software, vol. 28(5), 2011, pp.92-94.

[10] GS1 Ireland, "Beef Case Study," Feb 2005, *www.gs1ie.org* (@Jan 2012).

[11] International Organization for Standardization. BS ISO/IEC 27002:2005 Information technology. Security techniques. Code of practice for information security management.

[12] International Organization for Standardization. BS ISO/IEC 27001:2005 Information technology. Security techniques. Information security management systems. Requirements.

[13] C. Knight, Traceability in the Food and Feed Chain: General Principles and Basic System Requirements, Chipping Campden: Campden BRI, 2009.

[14] National Institute of Standards and Technology. National Vulnerability Database v.2.2, *nvd.nist.gov* (@May 2012).

[15] A. Nelson, "Embarking on an Outbreak Investigation," FOCUS on Field Epidemiology, vol. 1(3), North Carolina Center for Public Health Preparedness, University of North Carolina at Chapel Hill, 2003.

[16] The Open Web Application Security Project, *www.owasp.org/index.php?title=SAMM_-_Security_Requirements[_-_1;2;3]* (@May 2012).

[17] Organization for Internet Safety. Guidelines for Security Vulnerability Reporting and Response v.2.0 01, September 2004, *www.symantec.com/security/OIS_Guidelines%20for%20responsible%20disclosure.pdf* (@May 2012).

[18] A.L. Reingold, "Outbreak Investigations – A Perspective," Emerging Infectious Diseases, vol. 4(1), 1998, pp.21-27.

[19] J.K. Stehr-Green, "A Multistate Outbreak of E. coli O157:H7 Infection," Instructor's Version, CDC Atlanta, GA, April 2002, *www.cdc.gov/epicasestudies* (@Jan 2012).

[20] World Health Organization, Foodborne Disease Outbreaks: Guidelines for Investigation and Control, WHO Press, Geneva, 2008.