MARIUSZ WIECHEC
ASSOCIATE DIRECTOR
HKA

Why is my infrastructure project delayed by software bugs?

When we think of big infrastructure projects, we tend to focus on roads, bridges, tunnels and buildings. Traditionally, infrastructure upgrades involved civil and structural engineering, with most of the project costs related to this scope. Also, typically, it was the structural scope that eventually landed on a cover of a marketing brochure and that captured the public's imagination.

The nature of infrastructure projects is changing, with the focus shifting from simply delivering structures to delivering comprehensive functionalities. There is little point in building stunning, world-class infrastructure if, for example, all trains remain stuck in a tunnel on the red light while passengers are crowding on the platform, unable to even call in late for work because no one thought about providing cell network service underground.

"Familiarity" bias
Although it may not always be obvious, construction professionals are human. And, as humans, we have an inherent bias in assessing the importance and risks of the tasks at hand. We tend to gravitate toward what we "know" and ignore what we do not, a trait that, for the purpose of this article, I am labeling as "familiarity bias." If you only have a hammer, everything looks like a nail. Likewise, if all you know are sticks and bricks, then the entire project will be about that.

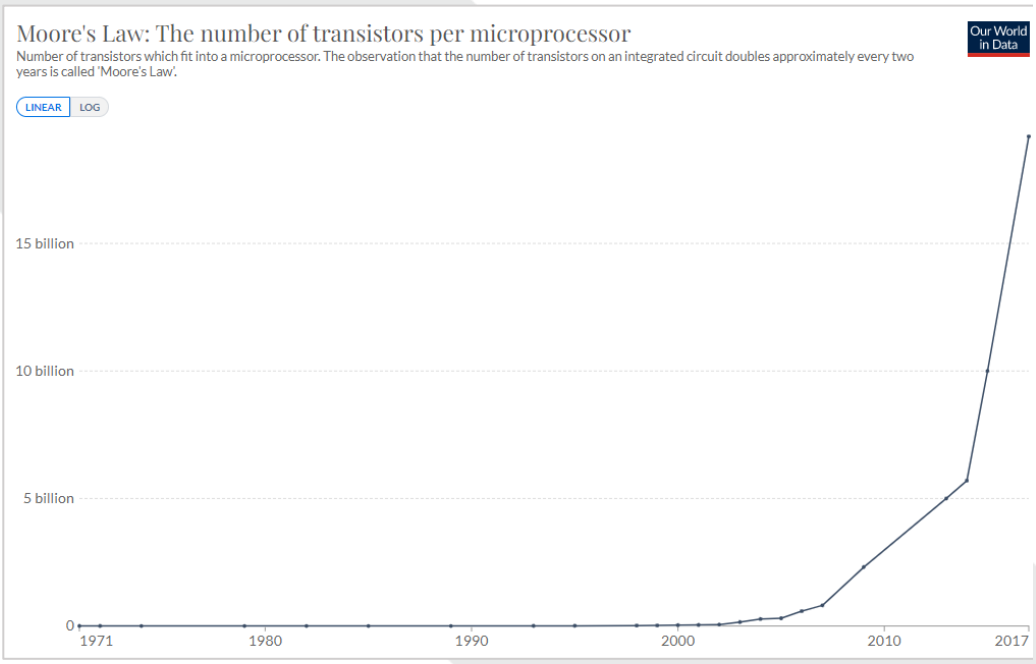
On modern construction projects, however, such familiarity bias can be limiting. Focusing solely on the structure often results in losing sight of development of all the supporting systems, which can be just as important to the project as the structure itself. And, this imbalance of focus often leads to project delays and cost overruns.

One of the components often overlooked in construction projects is the development of software for the safe and smooth operation of the infrastructure. **Software development is becoming an increasingly crucial component of construction projects, and such innovations as "smart" infrastructure, driverless vehicles, and intelligent transportation systems will only become more prevalent in the future**. It is important to recognize, however, that the process for software planning and development differ significantly from that of the typical construction process.

The rate of technological progress
Imagine a major railway project that takes ten years to complete, from planning through operation. A primary component of the project is the development of new software for the safe operation of trains. At the project's inception, the parties agree on a set of requirements and specifications for all the project elements, including operating software. By the time the project is finished, the technology available for software developers may be drastically improved versus what was available when the project started. This reality is consistent with Moore's Law.[i] Moore, the co-founder of Intel Corporation, in 1965 predicted that the number of

transistors on integrated circuits would double each year. As shown on the chart below, his prediction was accurate.[ii] Various studies of technological growth show that this exponential pace of progress is also true for other features of computer engineering.[iii]

Figure 1 – Moore's Law – Exponential growth of transistors in microprocessors illustrates a broader truth about the technological progress of computer engineering.[iv]



This continuous innovation is different from civil and structural engineering, where it can be safely assumed that the technology used today will likely be very similar and fit for purpose ten years from now.

The exponential growth of computer technology necessitates a different approach to managing software development projects. That is why more agile and less change-averse project management techniques, such as SCRUM (an iterative software product development process) are omnipresent within the IT world.

Unfortunately, these techniques also are at odds with a more rigid, traditional project management approach to construction projects, in which change is often difficult to manage. This discrepancy between the rapidity of technological evolution and the rigid approach to project management makes it challenging to manage projects that meld both civil/structural and software development aspects. And, because major infrastructure projects are very often run by professionals with a traditional construction management background, supporting systems such as software may not be foremost in their minds — or in their project documents and delivery plans.

Typically, integrating new software into the entirety of infrastructure occurs toward the end of the project timeline. The early stages of the projects are dominated by heavy civil engineering activities, with software development happening concurrently in the background. Thus, the negative effect of "familiarity bias" is enhanced by "proximity bias"—

focusing all attention on the task at hand, without consideration for what comes next.

This combination of factors may very easily lead to misalignment of priorities or a lack of attention to software development, which can result in delays and cost overruns. Moreover, this lack of attention to, or lack of understanding of, the dynamic and iterative nature of software development can lead to scarce documentation of issues, which in turn poses challenges when attempting to analyze delay or cost impact related to software development.

Crossrail – software delays

Crossrail is a railway construction project located in London and its outskirts. For more than a decade, it was a flagship infrastructure project in the UK. When construction began in 2008, Crossrail was touted as the biggest infrastructure project in Europe, with an initial budget of roughly GBP 17 billion (circa US$23 billion).

In its first few years, the Crossrail project was centered around the excavation of tunnels. Everyone understood that 26 miles of tunneling in central London would be an extremely challenging engineering feat, with BBC documentarians calling it "Urban Heart Surgery."[v] Tunneling works started in 2012, and while the 2014 National Audit Office (NAO) Report highlighted potential risks to project delivery, the overall tone of the messaging was positive, highlighting that: *'Crossrail Limited has achieved most or all milestones set for each six-month period on time."* The report further stated that there was an estimated 78-percent probability of Crossrail's central section opening on time in December 2018.

Five years later, the tone of the NAO Report was less positive. Although impressive and by no means easy, tunneling did not prove to be the biggest challenge. Tellingly, the project started experiencing significant delays in 2015, when the tunneling scope was finished. Multiple problems plagued the delivery. However, the perceived critical delays were still related to infrastructure development (constructions of stations and the tunnel lining).

To mitigate these delays, the project team decided to overlap construction with dynamic testing. This put enormous pressure on the delivery team. The testing could not be conducted efficiently, however, as it was discovered that "*the train and signalling software had not been developed to the required level."*[vi] Due to software development issues, *'few meaningful results could be acquired,"* and this unsuccessful testing effort *'took any spare time and space from construction workers on-site,"*[vii] which caused further delays and cost increases.

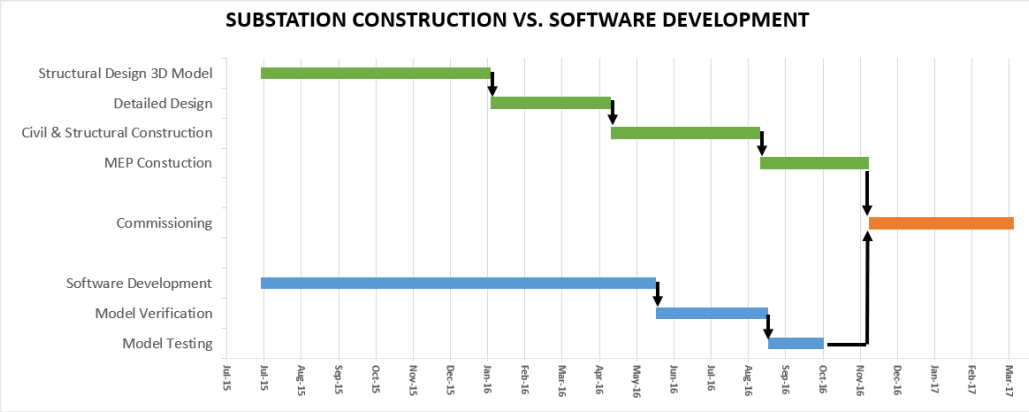
It is important to note that, even if the project team had delivered the infrastructure scope on time, the underdeveloped and delayed software would prevent successful testing of the entire system. While the software development was not the most significant issue causing delays to the delivery of Crossrail, it definitely played a role — and an unexpectedly significant one. Recently, Mark Wild, current CEO of Crossrail, stated in an interview, *'Crossrail is immensely complex, digital asset (…) more akin to a*

big IT program rather than infrastructure program." [viii] It was not always perceived as such.

Electrical substation – what is truly critical?
Projects do not need to cost billions of dollars to experience debilitating software issues. For example, a national electricity company hired an experienced general contractor to design, supply, deliver, install and commission new equipment for a substation to support area power infrastructure improvements. Apart from its civil, structural, and mechanical scopes, this $20-million project also included the development of software to operate the newly installed substation equipment.

The project's critical path delineated a fairly typical construction scope: Design → Civil and Structural Construction → Mechanical and Electrical Scope → Commissioning. However, there was also another, separate path, which involved the development of operating software to be installed on control panels in the substation. The two paths converged at the commissioning stage.

Figure 2 – Construction and Software Development paths (Illustration by Author)



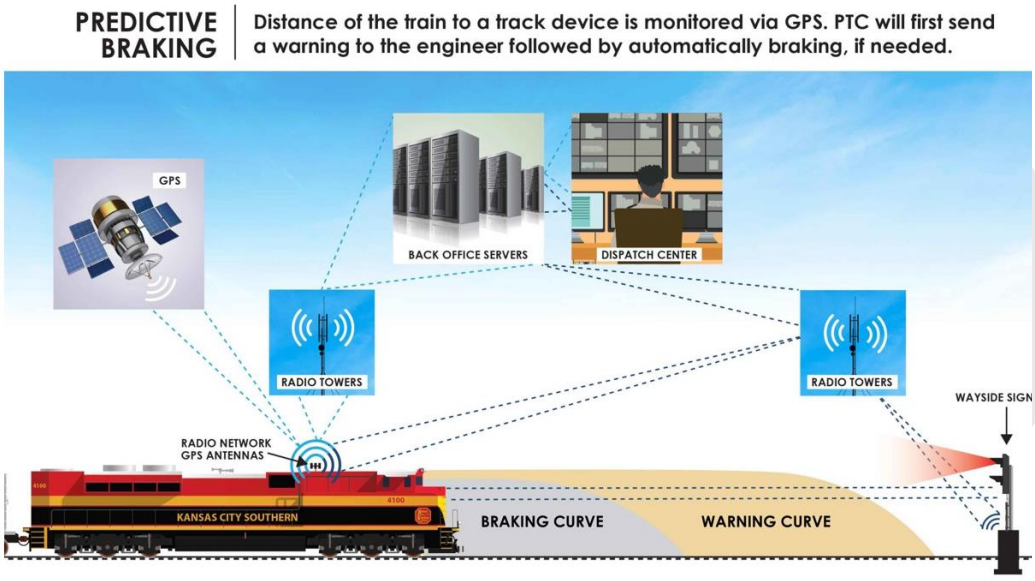
As in Crossrail, for the first couple of years of the project, the project team focused on developing the civil scope while the software development was progressing in the background. For this project, the owner was responsible for providing a software architecture environment (or a network model) within which the newly developed software was to operate. Ultimately, a lack of timely provision of well-structured software architecture became a root cause of the delays on the project. Had the software development scope been better understood and received more attention from both the owner and the contractor earlier in the project, the completion of the works likely would not have been delayed. Instead, severe problems with software development did not become apparent to the broader project team until shortly before commissioning, when it was too late to create a viable mitigation plan.

Positive Train Control System – can the software development be managed?
In 2008, the U.S. Congress passed the Rail Safety Improvement Act of 2008,[ix] requiring all Class I railroads' main lines that transport either hazardous materials or commuters to implement a Positive Train Control (PTC) system. Positive Train Control is a communication-based and

processor-based technology that is designed to prevent train collisions, derailments, and other types of train accidents. Initially, the Act specified that PTC systems be fully implemented by the end of 2015. Congress extended its deadline to 2018, and then to 2020, when it became clear that most rail agencies needed more time.[x]

PTC implementations involve multiple stages: installation of wayside equipment/hardware along the tracks (including wayside interface units, communication base- station towers, signaling and highway crossing equipment, and fiberoptics), onboard train control equipment and computer systems, and back-office control center servers and equipment. PTC installation also requires a multi-stage testing and safety certification process. Notably, the installation necessitates the development of software allowing for automated train control operations and "communications" between the various components of the whole system.

Figure 3 – Schematic representation of PTC system's major components and interconnections.[xi]



As with the Crossrail and Electrical Substation projects, PTC developments on railway systems in the U.S. have experienced multiple delays with similar root causes.

In its 2018 report,[xii] produced two years before the final PTC completion deadline, the U.S. Government Accountability Office (GAO) cited software issues as one of its concerns, but not the primary one. It is important to note that, in 2018, many railroads were at the early stages of their PTC testing program and had only begun to uncover the extent of software issues at this converging point. According to the report, *'nine passenger railroads reported encountering challenges related to maturity of the PTC software systems, such as working through software bugs or defects during testing."*

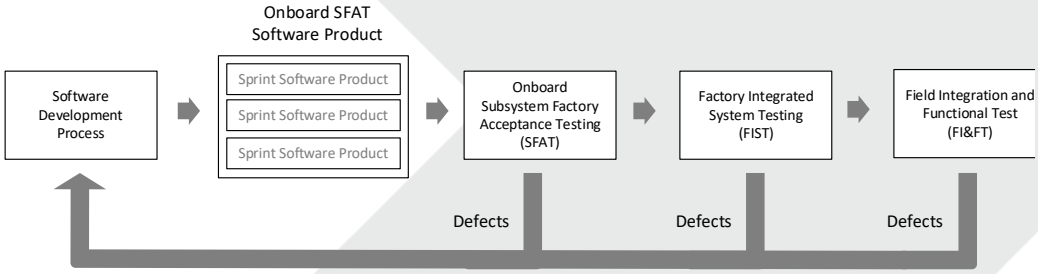
A year later, the updated GAO report[xiii] paints a different picture: *'31 of 37 railroads said software issues were a major or moderate challenge."* As the testing progressed, many project owners discovered that the PTC software was not ready for the subsystem and system testing. Resolving

the issues took longer than anticipated. *'Software issues are more acute now because as the 2020 deadline nears, less time remains to address these issues and associated delays,"* the report states.

As described earlier, the rate of technological progress for computer sciences vastly outpaces other engineering sectors. Of course, the railroads, the owners of the PTC upgrade projects, understandably wanted to utilize the newest available technologies. However, as the software was constantly evolving, the potential for scope creep increased and the appropriate management of the software development became cumbersome.

Moreover, as opposed to traditional linear planning of construction scope, the software development process is, by its nature, cyclical and non-linear.
Each functionality needs to be developed, tested at a product level, debugged and retested. This process repeats until the software is stable and functional. The same process is then repeated at higher levels of testing, including at the subsystem, system and field levels. New defects can be identified at each stage, which triggers further debugging and retesting.

Figure 4 – Cyclical process of onboard software development and testing process on a PTC project. (Illustration by Author)



For many PTC projects, in fact, the completion of software development has turned out to be a "moving target." It has become apparent that without clearly defined functionalities for design freeze and a well-developed change procedure that takes into account differences between agile and traditional project management and planning methods, software development can quickly become a significant issue.

In all three examples above, the lack of understanding or effective management of the software development process contributed to the project delays. With a heavy focus on the more tangible and typical construction tasks in the early stages of projects, the software development scope did not appear to become a top priority until much later in the project. In each of the cases, software issues became a point of discussion only at the testing stage (or shortly before) when project teams had limited opportunities to mitigate impacts.

In the interviews conducted for the GAO 2019 report, some railroad representatives stated that *'they had no control over this process* [software development]*, as they must rely on the vendor to provide reliable software."*[xiv] It appears as if a railroad's management team yielded

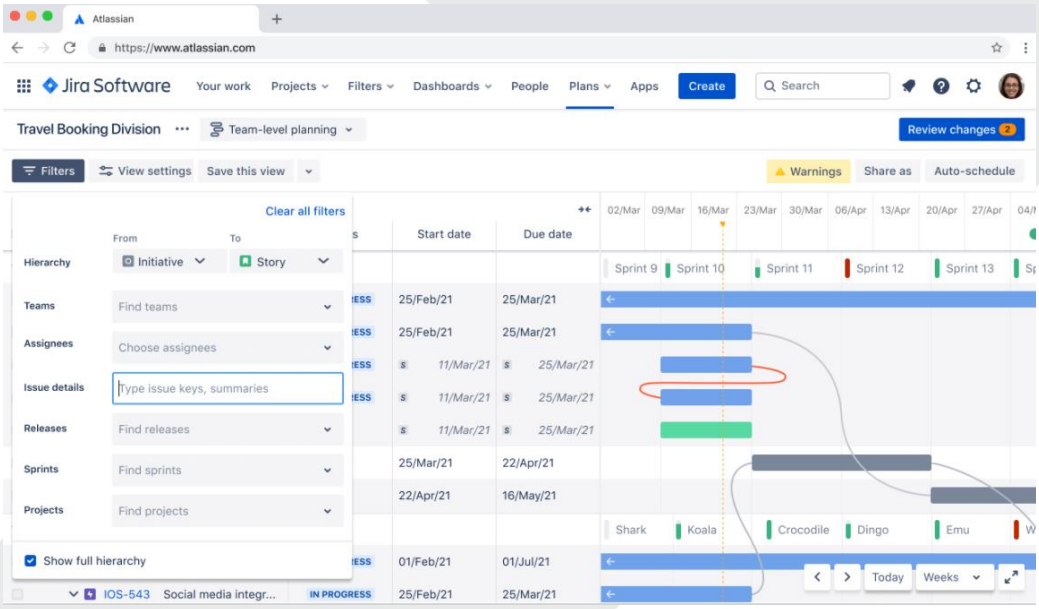
responsibility for controlling or managing this aspect of the project. The report further suggests that *'Representatives from this railroad also noted that resolving software issues is often not entirely within a railroad's control due to the need for vendor support, in contrast to some earlier challenges leading up to the 2018 deadline, where, for example, the railroad itself had more control as it was installing equipment and could more clearly track progress."*[xv]

Infrastructure Delivery – think systems, not structures

Software development should be managed, controlled, and analyzed with the same commitment and attention as any other key aspect of infrastructure development. So why is it that, so often, infrastructure projects fall short in that way? And, how can you prevent software development issues from derailing your project?

- **Consistent, Interdisciplinary Planning and Scheduling.** It often seems as if software developers and construction professionals speak different languages. This becomes apparent in discussions about scheduling concepts. "Gannt Charts" and "finish to start logic ties" might not mean much to a software developer, just as "product roadmaps" or "blockers" may not sound familiar to a construction scheduler. However, the reality is that some of the concepts are very similar. In fact, the terms above mean *more or less* the same thing.

Figure 5 – 'Roadmap' developed in a popular software development tool – JIRA[xvi] *heavily resembles construction projects 'Gannt Charts."*



To bridge this gap, the software development plan must be well represented in the overall project infrastructure development schedule. The schedule also should account for a cyclical, non-linear approach to the software development process. The planner should carefully establish a planned duration of this process considering risk exposure levels. This can only be achieved if the planners and schedulers have a sufficient understanding of the processes underlying the development and integration of software.

Also, the level of detail should allow for successful progress tracking of all tasks. Far too often, overall project schedules represent software tasks as single milestones (*'Software Delivered on date,"* for example, does not allow for good tracking of the progress). A good representation of software development progress in the project schedules also can be invaluable if delay analysis is needed. Most of the recognized delay analysis approaches rely heavily on the critical path method (CPM) and use of the official, approved schedule updates. If the software development is poorly represented in a project schedule, proving that it caused a critical delay to the completion may be very cumbersome.

- **Early Reporting and Data-Keeping.** Software development needs to be a part of project discussions early on. As discussed before, the functional software is typically only required at the latter stages of an infrastructure project, typically at the testing stage. With software being developed concurrently with more traditional construction tasks, and given the typically chaotic nature of construction projects, it is easy to ignore the software development scope. By the time the software is truly needed on a project, it is usually too late to mitigate delays. Therefore, much more attention and progress reporting are required early on. For example, software development can have a dedicated section in Periodic Progress Reports to highlight both progress and issues that need to be addressed. This reporting also can be crucial in a claim scenario when good project data is paramount.
- **Efficient Risk Management.** Risk Management efforts on construction projects often fall short due to "proximity bias." We tend to focus on what's right in front of us, rather than think beyond that and incorporate into our analysis things that aren't currently on our "to-do lists." As mentioned, functioning software is often needed toward the end of the project; therefore, it often takes time before it lands on the "to-do list" of decision-makers. By the time it does, risk mitigation options are limited. Therefore, project teams should consider software development risks early in the planning process.

It is important to note that all recommendations and suggested improvements will be fruitless without a major "re-think," coupled with a change of approach in how infrastructure projects are managed. In today's digital world, the public interacts with infrastructure differently, and its expectations of infrastructure are different. In fact, integrated, smart digital deliverables are often more important for end-users and owners than a "structure." To address related risks, the focus of construction management needs to shift from traditional civil, structural and mechanical scopes to broader, all-encompassing delivery of functioning systems as part of smart infrastructure. The UK's Institution of Civil Engineers recognized the issue in its new recommendations:

"*The dominant leadership and delivery model for infrastructure projects has not evolved to reflect these profound changes. Delivery remains in the hands of traditionally trained engineers working within organisations using long-established construction industry methods. The consequence of this conservatism is an increasing number of signature projects that are*

delivered behind schedule, beyond the cost estimate and that fail to meet the public's expectations."[xvii]

To meet the current challenges of a complex, technology-driven project environment, the infrastructure sector needs to adjust and adopt a new *systems-based* approach to delivering projects. Project management of infrastructure projects also needs to become more agile and apply a continuous development mindset similar to that so prevalent in software industries.

Infrastructure project narratives should be about project outcomes and providing a service to the public — not edifices. Every project is a complex system. The "sticks and bricks" scope should no longer be the sole focus of infrastructure upgrades; rather, all system components should be treated as crucial to overall success, with software development high on the list. As construction professionals, we need to focus on properly managing software development risks as an integral component of every infrastructure project.

If you require any further information, please contact Mariusz Wiechec at mariuszwiechec@hka.com.

---

[i] Gordon E. Moore - "Cramming more components onto integrated circuits", 1965
[ii] Karl Rupp – "40 years of Microprocessor Trend Data"
[iii] https://ourworldindata.org/technological-progress
[iv] https://ourworldindata.org/technological-progress
[v] The Fifteen Billion Pound Railway - Urban Heart Surgery - https://www.youtube.com/watch?v=IrLNJi2Zffs&t=11s&ab_channel=EngineeringProjects
[vi] NAO 2019 – Completing Crossrail – May 3, 2019
[vii] NAO 2019 – Completing Crossrail – May 3, 2019
[viii] The Smith School – "Are rail projects a nightmare to deliver? Lessons for successful delivery" – September 14, 2001
[ix] The Rail Safety Improvement Act of 2008, Pub. L. No. 110-432, div. A, 122 Stat. 4848 (2008)
[x] GAO-19-135T – Testimony Before the Committee on Commerce, Science, and Transportation, U.S. Senate – Positive Train Control - October 3, 2018
[xi] https://www.kcsouthern.com/en-us/ship-with-us/safety-security/ptc

[xii] GAO-19-135T – Testimony Before the Committee on Commerce, Science, and Transportation, U.S. Senate – Positive Train Control - October 3, 2018

[xiii] GAO-19-693T - Testimony Before the Committee on Commerce, Science, and Transportation, U.S. Senate – Positive Train Control – July 31, 2019

[xiv] GAO-19-693T - Testimony Before the Committee on Commerce, Science, and Transportation, U.S. Senate – Positive Train Control – July 31, 2019

[xv] GAO-19-693T - Testimony Before the Committee on Commerce, Science, and Transportation, U.S. Senate – Positive Train Control – July 31, 2019

[xvi] https://www.atlassian.com/software/jira/features/roadmaps

[xvii] Institution of Civil Engineers – "A systems approach to infrastructure delivery"