CS500215

Structured curiosity: the keys to unlocking efficiency within your process

Hamzah Shanbari
Haskell

Learning Objectives

- Understand the need for evaluation of technology and workflows as the industry and your organization evolve.
- Learn how to design and implement evaluation processes to empower decision makers and generate buy-in.
- Unpack Haskell's evaluation of Pype Closeout to see their process in action and discover what benefits they found.

Description

Learn how Haskell, a global Design-Build firm, harnesses their team's curiosity into an agile process evaluation that allows them to gain actionable insight from their data, effectively communicate it to stakeholders, and make strategic decisions that improve efficiency across their projects. Explore how a well-defined company mission and vision are the driving force behind making technology a cornerstone of the way Haskell works. Take a deep dive into the specifics of a pilot and evaluation process that provides both qualitative and quantitative data and see how that data is leveraged at scale across the organization.

Speaker(s)

Hamzah Shanbari
Manager – Construction Technology & Innovation at Haskell
Hamzah Shanbari has a strong background as a consturciton project manager and virtual design & consturcion specialist. Having a PhD. in Design, Construction, and Planning from the University of Florida, Hamzah has led Haskell's VDC team since 2016 and Dysruptek since 2018. He has published several academic papers on using Technology in Construction and Education in publications such as laser scanning, augmented reality and gamification.

The need for an evaluation process

Like many other global industries, AEC is moving rapidly towards a digital future. The innovations that lead to new tools and technologies for construction professionals to utilize are unfolding at a break-neck pace, driving efficiency, safety, and revenue gains for those companies willing and able to take advantage of them. Yet, for many organizations it is a daunting task to sift through these technologies to find what they can apply to their strategy and workflows, and what would be a bad fit.

However, simply sitting back and waiting for the best solutions to rise to the top isn't an option, either. At this pace, companies are increasingly feeling the pressure from competitors and clients to innovate or be left behind. This raises the stakes on their already difficult evaluations of technology to potentially add to their toolkits, as any misstep can cost valuable time, money, and buy-in from their employees.

Alignment

These missteps can often be the result of a mistake before the evaluation process even formally begins: trying to solve for the wrong problem. This is why many companies, like Haskell, continuously refer back to their mission statement and values throughout the evaluation process. Even just using the language found in those statements can help evaluators stay aligned with the rest of the organization as they define their goals for the evaluation.

The other half of that definition, however, should be informed by the problem statement. The problem statement is a simple, one sentence explanation of the specific pain point that the evaluated technology is supposed to solve. These pain points can be found in every job by every employee. Even those workers whose satisfaction is through the roof will have one or two processes that they wish would be less tedious, less complicated, less time consuming, and just less painful.

While this means your organization always has the opportunity for continuous improvement, it can lead some teams to overload paralysis—with so much to do, where do you even start?

Prioritization

Overcoming this paralysis can be done by seeing the forest through the trees. Mapping out these seemingly disparate pain points along their processes and among stakeholders can reveal larger patterns and point to shared issues. This is where your organization can start looking for ways to improve, as common issues are often easier to find champions for than single point problems.

And, finding champions for those issues is half the battle to generating buy-in. Not only are they invaluable for understanding the problem, as they have first hand experience with it, but they will also help you test potential solutions and provide feedback on those solutions. Without them, any evaluation process is all 'in-theory' which may not hold up to practical application.

But, what happens if you still have too many pain points of too high impact to effectively prioritize? The answer is simple: let them come to you.

The evaluation pipeline

The evaluation process is usually slower and more methodical than most would care to admit due to the high cost of making a mistake. However, this means that generating the essential buy-in during the process can be extremely difficult, especially when your pilot team just wants to get their work done without having to learn a new tool, cut through extra red tape, and fill out feedback documents on something they will possibly never use again.

Leads

That's why the last step in the prioritization of pain points is the same as the first step in an effective evaluation process: follow your leads. Whether that's a frustrated employee asking for help finding a better workflow for a daily task, a colleague talking about a tech start up he saw at a conference recently, or a recommendation from somewhere or someone else, following leads that approach you will be more time effective than searching around for a problem to fix.

Demo call

After you've identified a potential solution, you'll want to get on a demo call with a representative for that solution. Case studies, websites, videos, and other content have a place in helping you determine whether a solution solves for the problem in focus of course, but they are no replacement for an actual demonstration of the solution. Being able to sit down with someone who understands the technology and watch it work in real time helps provide a level of personalized insight that is priceless to have before agreeing to fund a pilot at your organization.

Internal discussion

Once you're as familiar with the technology as you can be, it's time to make a crucial decision in the evaluation workflow: do you proceed with a pilot or not? This decision should not be made alone, as this determines whether or not your organization starts to put money behind this technology. Instead, consulting with your champions, colleagues, or the representatives for the solution you're considering can help shine a light on any lingering questions you have about it's potential.

Additionally, before committing to a pilot of the solution, you need to have your pilot team or pilot project lined up. Your champions will be essential for this, as they are more often than not part of the pilot team or project. Otherwise, you may be committing to purchasing a license for something that will expire before you get a chance to give it a fair test.

Pilot

Your pilot process will depend heavily on the workflow and technology you are choosing to test. However, any pilot that you want to derive insightful data from should have an implementation plan in place beforehand. This document should outline all of the necessary information for your pilot, including but not limited to:

- what you are testing
- how it is being tested and by who

- who the other stakeholders in the pilot are
- what are your markers for success
- the prioritization of those markers for success
- roles and responsibilities of those involved

Having this document written and shared with your team ahead of time cuts down on miscommunication, unrealistic expectations, and uncertainty into the results of your pilot.

Analysis and report

Throughout the course of the pilot, you should be maintaining contact with the pilot team, champions, and other stakeholders. Collect their feedback constantly, including any effects the piloted technology has on other aspects of the workflow or their job. Once the pilot is finished collect the final statistics from the completed work, and compile this with the mid-pilot feedback into qualitative and quantitative categories. This will make it easier to compare your results to your success criteria and problem statement from the beginning of this process.

Once the results are in and the comparisons are made, you should have a pretty good idea of what next steps to take. However, to communicate that effectively you still need to compile them into a consistent report. Not only should this document include the qualitative and quantitative data collected from the pilot compared to your success criteria, but it should also conclude with a recommendation for action. These can include a wide range of actions, but should also always include a "why," for example:

- The technology meets or exceeds our expectations for success. However, the pricing structure being offered is unrealistic for what the technology can provide. Therefore, we recommend not moving forward with the technology until the pricing structure can be renegotiated.
- The technology does not meet our expectations for the pain point we identified, however we see potential for it to improve a different, specific workflow. Therefore, we recommend a new evaluation process of the technology for that pain point.
- Due to circumstances outside of the pilot team, the results from the pilot were inconclusive and unreliable enough to make a large scale decision. However, what results were received still indicate the technology's potential to solve this pain point as high. Therefore, we recommend additional pilot projects with additional parameters in place to avoid more inconclusive tests.
- The technology met our success criteria, and our pilot teams were very enthusiastic about it's use. The pricing is reasonable and the company offering the technology is stable. Additionally, the technology already integrates well with the majority of our tech stack, and the company offering the technology has indicated their willingness to work with us on future integrations. Therefore, we recommend moving forward with company wide implementation of this technology for this workflow.

Once your report is complete, ensure you share it with all stakeholders, champions, decision makers, and the pilot team itself. Doing so helps improve transparency and keeps everyone on the same page regarding the technology, helping to generate buy-in should the decision come to implement the technology, and if rejected, helping fans of the technology understand why.

Case study: Haskell's evaluation of Pype Closeout

The above process for evaluating technology was the framework that Dysruptek has used to assist Haskell in identifying innovations to solve their pain points. One such technology that underwent this evaluation process successfully is Pype Closeout.

What is Pype Closeout?

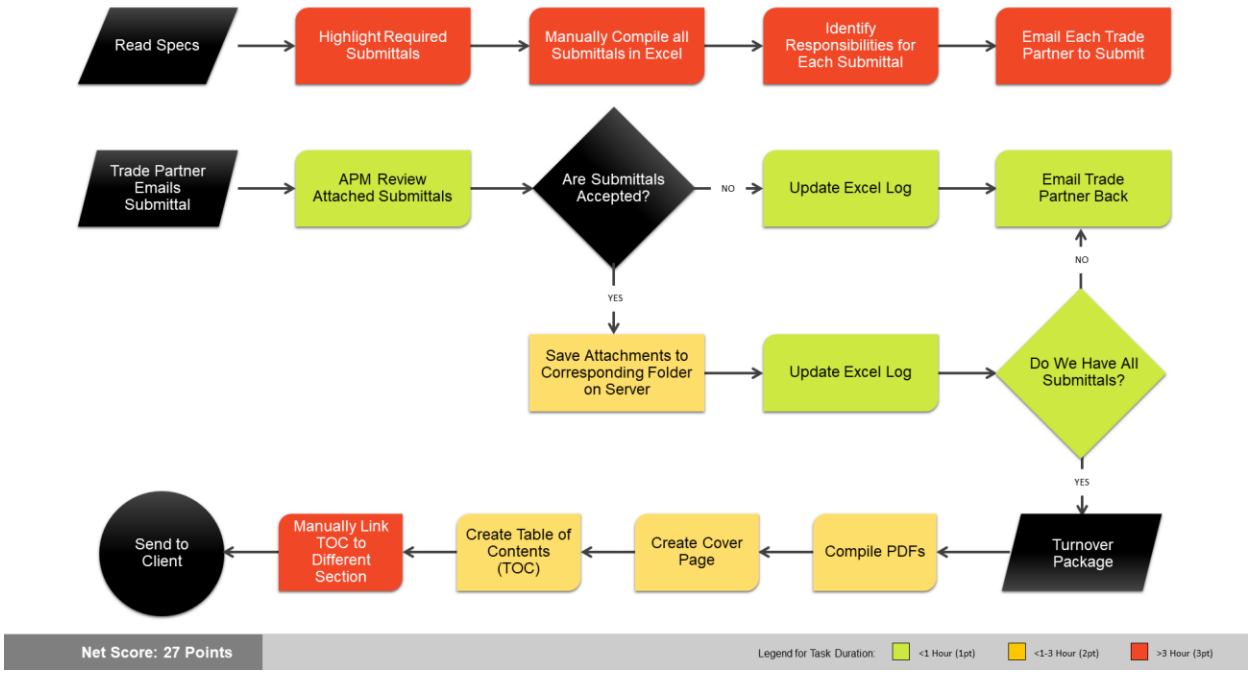
Pype Closeout is a web-based platform that automates the document closeout process. Users upload their project spec books to have the software automatically extract closeout requirement information. Alternatively, users can import closeout requirements from their project management software. With these requirements in the system, users can assign them to trade partners to submit against, who will be notified of their requirements via an automated email cadence with an adjustable frequency of notifications. These emails include a link for trades to upload documents against each requirement, where they will be stored in the cloud for users to review. Approved documents then move on to be compiled in the platform's turnover package generator. The simple user interface allows users to organize the documents in their package; customize their cover page, slip sheets, and table of contents; or conform their package to their organization's standards with company templates. These generated turnover packages feature an internally hyperlinked table of contents to prevent links leading out of the package from breaking, automatically placed slip sheets and page numbers, and are exported as PDF documents that can be stored in the client's cloud or printed.

What is Pype?

Pype was a technology start up founded in 2014 with Closeout, before quickly adding their second flagship product, AutoSpecs, which automatically generates submittal logs from uploaded spec books. The company expanded their offerings to include SmartPlans, which automatically extracts requirements from drawings, and eBinder, which is a stand alone turnover package generator. They were acquired by Autodesk in 2020, and are a standalone offering in Autodesk Construction Cloud.

Problem statement

The closeout process has far too many moving parts to make efficient. From coordinating with trades and the design team, reviewing hundreds or thousands of documents and submitting them or kicking them back trades; keeping account of what status each submitted document and each requirement is in as well as who is responsible for each; compiling approved documents into a turnover package that's usable for the operations team. We mapped out the entire process with the APMs, and assigned average work hours for each task. This served as the basis for our success criteria: be more efficient than this.



Haskell's preexisting document closeout workflow, assuming 100 documents 50% of which require one revision each.

Assuming an average of 100 documents per project, and that 50% of those documents would require one revision, the current document closeout process came out to take 670 hours of work, or 84 business days, from extracting requirements from the specs to generating a turnover package.

Pull, not push

A number of assistant project managers (APMs) had approached Dysruptek about the pain points involved with balancing document closeout on top of the rest of their responsibilities towards the end of the project lifecycle. After some exploring potential technologies, we reviewed Pype Closeout with the APMs and other key stakeholders, and decided the technology looked promising enough to proceed with a pilot.

The pilots

Ten Haskell project teams voluneteered to pilot Closeout as they each entered the closeout phase sometime over the course of the year. They were set up with onboarding and implementation from Pype's own customer success teams, and channels of communication between Dysruptek, Pype, and the pilot projects were set up to both collect feedback and make sure the projects did not suffer should the technology fail to meet success criteria. Additionally, as the pilots move forward, three more projects wanted to pilot the software as well, bringing the total pilot projects up to 13.

Initial results

The pilot teams started to provide usable feedback during the third onboarding call. They felt that the closeout requirement log the software had generated was very accurate, although it still needed someone to pare down some extraneous requirements. However, they said this method was more timely than our current workflow, which meant they could deliver the logs to trades sooner and in turn collect documents from trades sooner.

Qualitative feedback

Throughout the pilot and after project turnover, both Haskell employees and the trade partners they worked with on the project were surveyed about their experience with Pype Closeout.
Of the Haskell employees surveyed:

- 90% were extremely happy with the software, and perceived that it allowed them to work more efficiently.
- 85% then recommended using Pype Closeout on future projects.

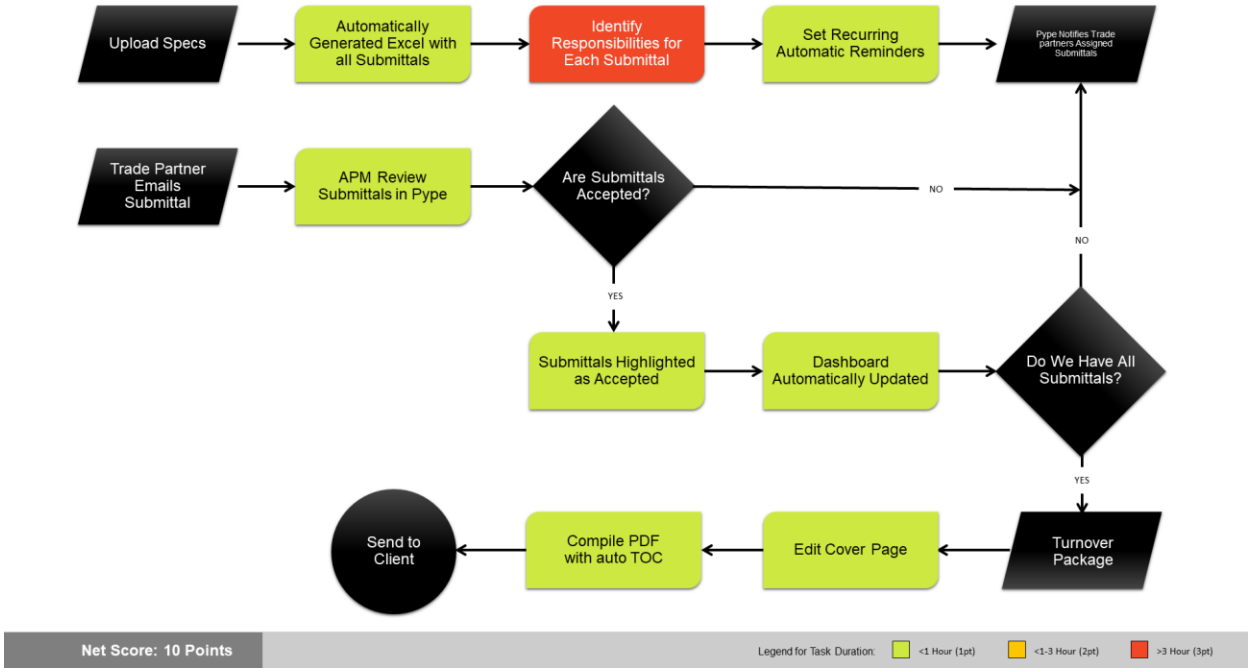
Of the trade partners surveryed:

- 93% perceived the assigned submittals as accurated based on the spec books
- 73% thought it helped them closeout their scope of work faster.
- However, only 46% of the trade partners went on to recommend using Closeout in the future.

Dysruptek thought this last metric stood out, and followed up with the trade partners who did not recommend future usage. Those trade partners felt that the software was simply one more thing for them to keep track of, especially with it’s own unique login information for them to remember.

Quantitative feedback

Based on the mapping of the current workflow and times reported by the pilot project team members, Dysruptek created a version of the workflow map based on using Pype Closeout.



Haskell's document closeout workflow using Pype Closeout, assuming 100 documents 50% of which require one revision each.

Again assuming 100 documents on average for a project and that 50% of those documents would need one revision, the document closeout process with Pype Closeout would take an average of 357 hours, or 45 business days, from extracting requirements from the specs to generating a turnover package.

By comparing the two maps, Dysruptek determined that Closeout saved the teams on average 295 hours of work, or about 39 business days, cutting the time spent on document closeut down by 46%. Additionally, using Closeout, the 13 projects collected 2,790 documents overall.

Conclusions and report

Based on the results form the 13 pilot projects, Dysruptek recommended that Haskell move forward with implementing Pype Closeout company wide.