

## How to Avoid Waterfalls and Achieve Better Results with Your Speech IVR

Most speech-enabled IVR application vendors use traditional waterfall development processes that don't capture direct user feedback until very late in the project. Because changes are expensive to implement at late stages of development, waterfall processes actually discourage full optimization of the caller experience. This whitepaper describes a unique application of Agile software development principles to the process of designing and developing speech applications. This process, created by PSS, provides for early and frequent cycles of user feedback to ensure the best possible caller experience. Better caller experiences yield better results, helping ensure the best possible returns from investments in speech applications.



## Would You Buy a New House Sight Unseen?

Probably not. But when you deploy a new speech-enabled IVR application today, most vendors expect you to take a similar risk.

Imagine spending weeks or months building a business case for your new speech project. You forecast business results and commit to an ROI attractive enough to get the project funded. You select a vendor and kick off the initial system design. Anxious to hear what your new application is going to sound like, you want to be sure you're on the right track to achieve the results you promised. How soon will that happen?

Once you commit to purchasing a new speech IVR system, it may be months before you experience a meaningful preview of what you bought. You may see call flows on paper during the design phase, but typically you will get almost no sense of the caller experience until very late in the deployment process.

This is a big risk because when you wait until the application is nearly complete before you incorporate meaningful feedback from users, you are committing to a design virtually sight unseen. This risk is inherent in most speech projects because the vast majority of speech vendors adhere to a traditional “waterfall” development methodology.

A waterfall project is strictly divided into cascading steps, and each new step doesn't begin until the previous step is complete. Requirements are gathered and documented at the top of the waterfall. Requirements documents flow down into design, design cascades into development, then into testing, and finally into deployment.

In spite of being the norm for speech projects, the waterfall method steers the majority of today's speech application development down a path that increases risk and discourages improvements to the caller experience. This paper examines a better process based on principles derived from Agile Software Development.



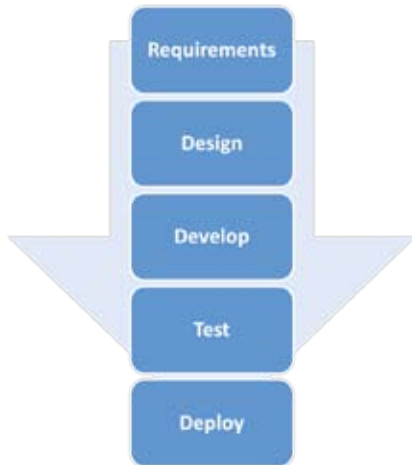
## You Can't Experience a Blueprint

The value of a well-researched, thoroughly tested Voice User Interface (VUI) is undisputed—better caller experiences yield better results and help ensure the best returns from your speech IVR investment. Without an efficient, effective, and likeable user interface, even the most technically sophisticated speech system will fail. In a waterfall speech project, VUI design is completed early in the process before developers begin writing code. At the end of the development phase a usability test is conducted to obtain caller feedback before putting the finishing touches on the nearly completed application.

The real question is whether the entire VUI design process must occur before you can move on to development. Good design takes time, but does it all need to happen up front?

The theory behind the waterfall development method is painfully simple. It's a very procedural insert-tab-A-into-slot-B sort of arrangement:

1. Fully define and document the requirements and scope of the project before you begin the design,
2. Fully specify and document the entire design before beginning development,
3. Complete the development before you move into testing.



According to waterfall methodology, you completely document a project before you start design so you can avoid trying to hit a moving target. First you define system requirements, then the vendor documents those requirements so both parties can agree on the scope of the project. At the end of the process, you get a detailed description of everything you intend to accomplish in your speech project- much like the blueprint for a house.

An architectural blueprint has two purposes: it documents detailed technical information that the experts need to build your house, and it gives you a preview of what your house will look like.

But blueprints have limitations. Think about how well a blueprint actually previews the experience of living in your house. From the blueprint you can tell that the kitchen is next to the family room, and you can tell there are three windows spaced evenly along the wall. But you can't really tell what it feels like to be in the kitchen.

On paper, three windows look good, but it's not until you actually experience the kitchen live and in person that you can tell that the kitchen seems dark. It's only once you experience what it's like to be in the kitchen that it becomes obvious that you need another window. Unfortunately, by the time you're actually standing in the kitchen of a nearly completed house, adding a window requires a significant investment of time and money.

Simply put, a blueprint doesn't reveal everything about a house—and a requirements document doesn't tell you everything you need to know about your speech application. In most speech projects, you're asked to sign off on your speech application based only on a blueprint that you won't see again until the whole house is built.

This limits your ability to incorporate valuable user input into the design process so you can be sure you're creating the most effective and desirable caller experience. By the time the house is built, it would cost too much to add that window in the kitchen, so you end up compromising your living experience. Likewise, with a waterfall speech project you don't get live user feedback until the application is already fully built. This is the most labor intensive and expensive time to make improvements, so you may have to settle for a compromised user experience.

*"An architect's most useful tools are an eraser at the drafting board and a wrecking ball at the site."*

FRANK LLOYD WRIGHT

## Better Process. Better Results.

There's a better way. PSS has developed a unique speech project methodology based on Agile Software Development principles. This process interweaves design, usability testing, and development across several project phases.



*Like walking through your custom home while it's being constructed, Agile Speech Projects give you ample time to make improvements based on actual user feedback.*

With each passing phase, the application gets built in layers as new improvements to design, further testing, and more development take place. Instead of having to rely on a waterfall blueprint, this allows you to continually experience your application first hand as it matures throughout the project- months earlier than waterfall speech projects allow. More importantly, frequent previews and usability test results help you identify improvements to the caller experience much earlier in the process- when it's still relatively quick and inexpensive to do so. In an Agile Speech Project you get many chances to make changes with an eraser instead of a wrecking ball.

### Get Agile.

PSS Agile Speech Projects proceed rather differently than waterfall projects. Instead of trying to build the entire application in a one-way progression of steps, we work in phases, evaluating and making adjustments as we go. At the end of each phase you can preview the customer experience as it matures.

We prioritize the highest value elements of the application—the ones you're counting on to make the ROI work—and tackle these early in your project. Each phase is like a mini project, with VUI design, end-user testing, and development included. Phases last for weeks instead of months, so you're able to experience the most important parts of your new speech application much sooner than in a typical waterfall project. But this isn't speed for the sake of speed—this is speed that actually builds in responsiveness to changes triggered by user feedback or even external conditions in your business.

No matter how carefully you plan for a speech project, inevitably you will encounter changes along the way. Good decisions you made at the start of the project sometimes become bad ones because conditions have changed and your original assumptions are no longer valid.

In a typical waterfall project, the vendor focuses on methodically documenting how you thought you wanted your speech application to work back when you started the project. Intuitively this may seem like a good idea, but in practice it's counterproductive. Forcing complete documentation up front makes it harder to respond to changes as they happen. Since everything must be documented before moving on to design and development, even simple updates trigger a change request process that must be reconciled with original requirements.

It's like adding that kitchen window once the whole house is built—usually you decide to just accept living with a dark kitchen rather than take on such a big new task when the project is so close to completion. Rigorously complete documentation of requirements actually discourages you from making just-in-time adjustments to your speech self-service strategy. In Agile projects, you get the chance to adjust your strategy in each phase.

*It's much faster and easier to add an extra window during framing than after a house is drywalled and painted.*

*Agile phases ensure that end-user testing is iterative and ongoing, thus increasing the amount and quality of actionable user feedback.*

This responsiveness also applies to changes based on end-user feedback. No matter how smart the experts are, we never know how end-users will react to an application. This is why we run usability tests—to find out what works for the end-users of your application in the context of your business. Usability testing should be an iterative process of discovery, design, testing, and revision, but most waterfall speech projects include only one or two usability tests. The usability testing typically occurs close to the end of a waterfall speech project when the application has already been fully designed and developed.

Unfortunately, it can be expensive to fix usability issues if you discover them too late. Too frequently you end up resigning yourself to a sub-optimal user experience in order to meet deployment dates and stay within budget.

The PSS Agile process builds end-user testing into every project phase, allowing us to continually refine the overall user experience. Agile phases ensure that end-user testing is iterative and ongoing, thus increasing the amount and quality of user feedback we can include in any given project. Because we collect user input throughout the project, it costs you less time and money to make changes that reflect users' opinions and desires.

## Agile Teamwork Gets Results

If you've been through a typical speech project, you probably need a reality check about now. Agile Speech Projects sound good, but how can we make this work? Can we really gather requirements, design, develop, and test a chunk of a real speech application in a matter of weeks? Most speech project teams would be hard pressed to deliver under such conditions.

On most speech projects, there are silos of specialization: VUI designers who craft prompts and call flows, usability specialists who interact with end-users, developers who write the code, speech scientists who write grammars and tune applications. Even if the different specializations all work for one vendor, they often don't communicate well. Have you ever heard a developer bemoan ambiguous designs or a VUI expert complain about developers who misinterpret their designs?

*Have you ever heard a developer bemoan ambiguous designs? Or a VUI expert complain about developers who misinterpret their designs? Agile Projects facilitate communication better than waterfall projects.*

The waterfall process encourages specialists to stay isolated within their own domain and to pay attention to only their project stage. For example, development is a project phase that begins only after design is complete, so developers have no incentive to be involved during the requirements or design steps. This leads to a mentality that encourages each specialist to complete his given task in his own project step, then "throw it over the wall" to the next specialist in line.

The specialists in a waterfall project rely on documentation to communicate everything they need to know from preceding phases of the project. Unfortunately, even the best documentation is imperfect. When developers have a choice of techniques to employ when coding a design, they may be unable to make the best coding choice if they are unaware of the requirements that motivated the designer.



When the development specialist is unsure about what the VUI specialist intended, there are two possible outcomes. The developer could stop coding and re-engage the VUI designer to try to make a good coding decision. This diversion costs the developer time and can be problematic if the designer is already engaged in another project. Alternatively, the developer may have to forge ahead without understanding the motivations behind the design. This increases the risk of making a bad decision. Similar situations occur throughout waterfall projects whenever you move from a step with one set of specialists into another step with a different set of specialists. Miscommunications and misconstrued intent are the inadvertent but inevitable result of the waterfall method. These problems can cause delays and reduce the overall quality of your speech application.

PSS Agile Speech Projects avoid these communication problems by organizing work in highly aligned project teams. Every team member is fully invested in the overall user experience of the speech application, so the whole team is engaged throughout the project.

Because Agile Speech Project phases are brief, PSS includes the whole team in each project activity. In waterfall projects this is impractical. Vendors typically can't afford to tie up developers for weeks while VUI designers collect user requirements, but because requirements-gathering happens so quickly in Agile Speech Projects, the Agile developers learn about user requirements first hand. Rather than relying solely on documentation to ensure that everyone is on the same page the whole Agile team participates and understands requirements, so each team member is able to make better decisions for their own piece of the project.

Agile Speech Project teams have multiple opportunities to re-align during each phase as well. Because we collect feedback from both the business and from end-users of the application in every phase of the project, we can evaluate the quality of the decisions we made in that phase based on a continuous source of high quality data. Each Agile team member sees what works and what doesn't, and is empowered to make changes within his specialized area to correct problems.

## A New Way of Thinking

The Agile method isn't just a rearrangement of the tasks and responsibilities of a typical speech project—it is a completely new way of thinking. Every member of the Agile team has the same goal: to create the best experience for your customers so you can achieve the best results with your speech application. Each team member shares similar experiences as they gain understanding of your business and user requirements, so each team member sees how their contribution is critical to the success of the project.

In waterfall projects, not all team members share a vision of the user experience for the application they're working on, so their decisions are not always guided by the same motivations. An Agile team must be closely aligned because they don't rely solely on documentation to communicate. The specialists on an Agile team can't stay in their silos—the process demands they create speech applications while collaborating in parallel, rather than working in sequence.

Working in parallel removes the inherent tension in waterfall projects of having to wait for someone to finish a previous task before you can start your own. It's not uncommon for different specialists on waterfall project teams to view each other as adversaries in competition for time and resources. This is a direct result of the way waterfall projects work: if the design takes five days longer than planned, development and all subsequent tasks start five days late. If the project completion date doesn't move, design has in effect stolen time from development. Agile Speech Projects remove this internal conflict by aligning interests among team members.

## Agile Speech Project Flow

So what does an Agile Speech Project look like? The real innovation here was figuring out precisely how to apply mainstream Agile Software Development principles to speech. Strictly speaking, mainstream Agile software development methodology isn't a good match for speech projects if you try to apply it directly.

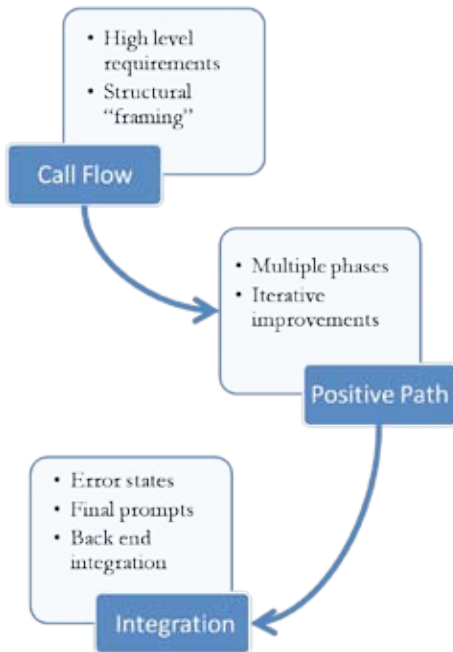
Agile software development reduces risk by dividing an application into functional blocks. Each block includes a quick design-develop-test cycle, so you end up building the complete application block by block. Agile speech development has a similar goal of reducing risk and uses a similar quick design-develop-test cycle. Unlike Agile software development, Agile Speech Projects don't necessarily generate a fully available release of the speech application at the end of each cycle.

It's relatively easy to develop a GUI interface feature-by-feature, because adding another item to a visually displayed menu does not have a large impact on the user experience. Mainstream Agile development works well with a GUI. But it doesn't make a lot of sense for speech, where each additional feature can completely change the user experience.

Presenting a new feature in speech adds time and complexity for the caller, which can affect the overall usability of the application. This happens because speech is linear (we can only say one thing at a time) and transient (there is no permanent record of prompts that the user can refer back to in a speech system.) So a direct application of Agile software development methods to speech projects doesn't make much sense. Speech applications must evolve in such a way that the user interface is gradually improved instead of continuously disrupted as each new feature is added.

## Call Flow Phase

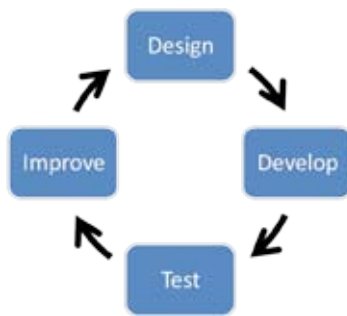
Instead of breaking an application into feature blocks, our goal is to develop the speech application in layers, adding complexity with each cycle. The first phase of an Agile Speech Project concentrates on the deepest layer of the application, the call flow. In this phase we capture the application state-by-state, defining what options the application will present to callers and what information we will collect from callers. This is the phase in which we map out communications between the IVR and backend databases in order to understand how data will move back and forth.



*Applications are built in layers, adding complexity with each cycle. This allows for application previews that enable earlier and more frequent direct user feedback.*

We tackle call flow in the first phase of an Agile project because the call flow represents the skeleton of a speech application. The call flow is like the framing for a house- the electrical, plumbing, and heating systems are all built overlaying the framing. Similarly, the rest of the application- prompts, grammars, business logic- all depend upon the call flow. It's easy to move a wall when the house has just been framed, but it requires demolition once you've run the wiring and put up drywall. By establishing the call flow and associated data transactions early, we minimize the possibility of demolition later in the project.

There are other benefits to approaching call flow as a distinct initial phase of the project. We engage the development team significantly earlier than in traditional speech projects, which allows them more time to understand both the structure of the application and how the IVR will interface with backend systems. Data transactions are notorious for causing problems in speech projects and the Agile process ensures that we uncover problems as quickly as possible. Focusing on call flow alone also lets us use techniques such as cognitive walkthroughs, user needs assessments, and task analyses to collect user feedback. These techniques are well established ways of understanding users' mental models for tasks, but they are typically omitted from speech projects because there is no time for user testing early in the project. These techniques enable VUI designers to gather additional data from end users early in the process so it's faster and easier to make improvements based on that input.



*Each Agile project phase includes cycles of design, development, user testing, and improvements.*

Finally, the call flow phase lets you, the customer, be more involved in the design and development of your application earlier in the project. Not only does this give you the opportunity to experience your new house in the “framing stage”, but it also gives you a better understanding of why design and development decisions were made because you can be present during the decision-making. No more wondering “How long until I can see something?” or feeling marginalized in the design of the application you're paying for.

## Positive Path Phases

Once improvements based on feedback in the call flow phase are implemented, an Agile Speech Project moves into the positive path phase. A positive path is a route through the call flow that proceeds as planned- i.e. no error states occur. In the positive path phases the PSS Agile team fleshes out the skeleton application, adding prompts, basic grammars, and detailed coding to the application. We focus on the actual sound and feel of the interaction from the user's perspective.

The size and complexity of your application determines the number of positive path phases we need. Applications with many branches typically have many possible positive paths, so the goal is to divide the full feature set into small enough chunks so that each phase has a manageable subset of features and positive paths.

Because the entire call flow has already been established, we can easily work on small chunks of the application at a time without sacrificing the overall user experience. Working on a few features at a time allows us to concentrate our efforts in one direction and gives you more opportunities to observe and react to your new speech application.

During the positive path phase we add business rules that determine how the application will handle specific callers in specific situations. We also script and record the prompts for each positive path during this phase. In this phase you will have the opportunity to select the voice talent who will record your prompts and establish the way the application will express the branding characteristics most important to your organization.

A centerpiece of each positive path phase is the usability testing conducted to gather direct feedback from live users who interact with a functional prototype version of your application. The prototype application uses real prompts and real speech recognition to give an accurate preview of the final application so we can get the best possible feedback from users.

*Usability testing with live users is the centerpiece of each positive path phase.*

The prototype applications do not yet include live connectivity to any backend databases and systems. This allows the VUI specialist to focus on users' reactions to the sound and feel of the application without any interference from issues related to backend systems. It also gives the development team additional time to perfect host and database connectivity. We include one usability test in each positive path phase, so you have multiple opportunities to observe users interacting with your application at a time when it is still quick and easy to make changes.

*These users interact with a functional prototype version of your application to provide direct feedback on the experience.*

*This feedback drives the cycle of design, development, and improvement at each phase that enables the Agile process to be so user-centered.*

## Final Phase: Full Integration

In the final phase of an Agile project we fully integrate the user experience with the technology that will support the application. By this phase of the project, the user experience has been finely honed via several rounds of design, testing and revision. Because so much user feedback has already been incorporated into the application, we minimize the possibility of encountering any major “structural” changes necessary at this late stage of the project.

Because we have already optimized the positive path user experience, now it makes sense to take the time to fill in all the details, such as scripting and recording prompts for error conditions throughout the application. We wait until late in the project to complete error-handling prompts because they are time-consuming and tedious to change. To return to our construction metaphor, we don't bother painting the walls until we know they're in the right place.

In this phase we integrate the user-facing elements of the application with backend systems and test again with end users to ensure that we maintain a positive user experience. We collect user feedback at this stage using field tests in which users interact with the system in their real calling environment instead of a lab setting. We still give users tasks to complete, as in earlier usability

testing, but we now observe the interactions in a realistic setting. Because we have already conducted iterative usability testing, issues we discover at this stage are likely to be minimal and related to differences between the lab and real world environments. Data from field tests also allows us to look at recognition performance and make adjustments to grammars and recognition parameters.

## Deploy with Confidence

At the end of an Agile Speech Project, you can deploy your new speech application with extreme confidence. You can be confident because:

- You already know how users will react to the application. You've already observed users reacting to the application multiple times.
- You have been involved in making decisions throughout the project and had numerous opportunities to provide your own feedback to the Agile team
- The Agile process ensures that your application has been optimized through multiple rounds of design, development, testing, and revision. The process allows you to adapt quickly to changes you encounter along the way, such as modified business goals or unexpected caller reactions.



## Get Agile. Avoid Waterfalls.

Waterfall speech projects force you to nail down all the requirements at the start of a project then penalize you for trying to make improvements as you go, because the first opportunity to make changes based on user input arrives near the end of the project- when those changes are most expensive and time consuming to implement. Because waterfall projects discourage optimization, in the end you usually settle for a caller experience that you know could have been better.

The unique Agile Speech Project methodology developed by PSS gives you early and frequent previews of your application as it grows in layers. The process has built-in cycles of user feedback, evaluation, and optimization so you can be sure callers get the best possible experience using your new application. The most successful applications are the ones callers are delighted to use. A better caller experience leads to better results from your speech application and help ensure you get the best possible returns from your speech IVR investment.

## About PSS

PSS helps enterprises transition gracefully from legacy to next generation contact center solutions. Through a unique combination of products, services, and 24x7x365 Extended Life support, PSS customers can transition to VoiceXML, CTI, VoIP, SOA, and Web Services as rapidly or gradually as their business priorities demand. PSS customers include American Express, US Airways, United Healthcare, and CapitalOne.



**PSS is one of the fastest growing  
contact center companies in America.**



### **Business Office**

7172 Regional Street #431  
Dublin, CA 94568  
925-208-2450

### **Logistics Office**

214 Ontario Street  
Frankfort, IL 60423  
925-208-2430

### **24x7x365 Support Line**

888-455-2285

### **UK Support Line**

0.808.234.6787

### **Sales Inquiries**

925-208-2450  
sales@psshhelp.com