Agile Methodology: The Future of Insurance Product Development?

By Kelly Rabin and Gary Baluta

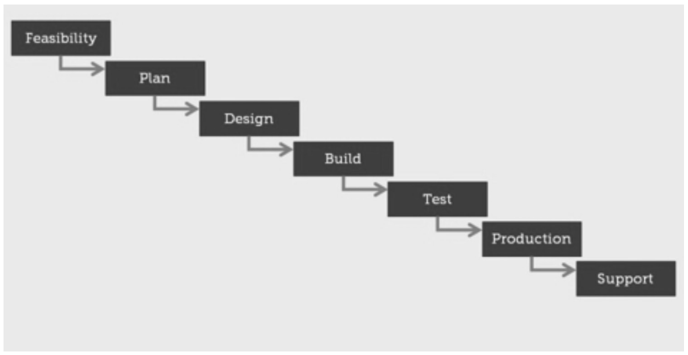
"Agile" development is all the buzz in the IT world. You may have heard your IT teams talk about "scrums," "sprints," and "the war room" and wondered if they were playing a strange new sport. Or perhaps you are already applying agile techniques in your product development. This article will discuss what agile methodology is, the pros and cons, and how you might apply it to insurance product development.

WHAT IS AGILE DEVELOPMENT METHODOLOGY?

Agile development methodology is a relatively young and much more collaborative way of creating and launching products to the marketplace, most commonly used in an IT setting. Before we get into the details around agile, we will discuss the more traditional approach it replaces, typically referred to as "waterfall."

What does a waterfall approach look like? Think Gantt charts and MS Project, as illustrated in Figure 1. It is a stepwise approach in which each phase of the development life cycle is completed by the appropriate team and then passed to the next team. The major issue with this approach is that it often limits communication between the various groups involved in the product life cycle (product management, development, and quality assurance), sometimes resulting in products that do not meet expectations. Formal requirements and specifications documents are created and approved early in the process. If there is minimal input and collaboration with downstream areas during this process, the specs can be subject to misinterpretation and unauthorized modifications.

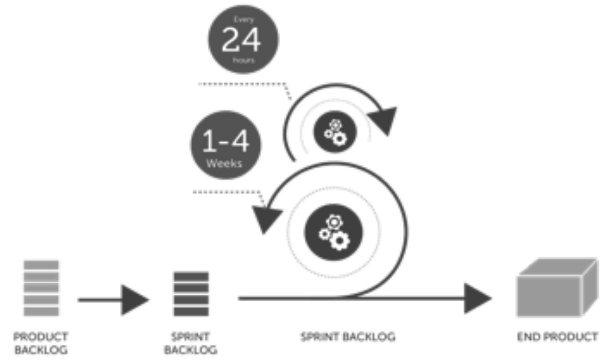
Figure 1. Diagram of Waterfall Development Methodology



Source: Manifesto Digital, London, England

How does agile differ from this? The core of agile methodology is that decisions are not made until they have to be, allowing the product to adapt to evolving business conditions. Rather than locking down specs for the entire build at the inception of the project, agile uses cycles called "sprints" that focus on specific pieces of the project (see Figure 2). Requirements are determined as needed. The length of a sprint can vary, but typically it is two to four weeks of focused effort. During the sprint, teams are ideally co-located (with access to a dedicated collaboration space called "the war room") and have daily team meetings (often called "stand ups" because they are intended to be brief and focused on communicating key open issues to the broader team and identifying next steps). This culture of open communication and collaboration aims to avoid "coding in a vacuum" in which a programmer is only focused on their piece of the project and is not aware of how their work might impact other areas. It also aims to avoid launching products that are already obsolete.

Figure 2. Diagram of Agile Development Methodology



Source: Manifesto Digital, London, England

A common agile approach is referred to as "scrum." There are three main roles in this process:

- **Product Owner** – a member of the product management team (or someone who works closely with them) responsible for defining the product and prioritizing the importance and order of the development task list (called the backlog). They are also the liaison to customers and internal stakeholders, responsible for identifying market needs and keeping everyone up to date on project status. The product owner's primary focus is on product content.

- **Scrum Master** – a member of the development team primarily focused on managing the processes required to successfully build the product. They also facilitate the exchange of ideas and ensure that the team remains organized and efficient.

- **Team Members** – analysts, developers, and testers who are responsible for the formal building of the product. There are no managers assigned to the team. Members get their task list (called stories) from pre-sprint meetings (called iteration planning) and are responsible for completing them as expected. They all have equal input into determining which tasks are to be worked on and by whom. Estimates for how long each task should take are also discussed and agreed upon before each sprint commences.

While the concept of agile development sounds fairly simple, it is often anything but. There are many moving parts inherent in this collaborative process. This can cause poor results if the product owner and/or scrum master are not plugged in to what the team members are doing. The nature of agile methodology can result in many decisions being made "on-the-fly" that often are not as well thought through as they should be. There is risk that developers will make important design decisions without consulting the product owner. Co-location and stand-up meetings are intended to reduce this risk, but they are not foolproof, particularly when on a tight deadline.

In order to address these issues, organizations have recently started to utilize a hybrid version of agile development in which a documented set of requirements is created before development begins. These plans are shared with the development team in an iterative review process, with the intent being to finalize the "what," in terms of product content. The development team then scopes out the details of the product design using the list of requirements previously agreed upon. These details are documented in the specifications, representing the "how," how a product should be coded and should function from a technical perspective. This is also an iterative process requiring discussion and consensus. Once the requirements and specifications documents have been created, the agile process kicks in—stories are defined using the existing requirements and developed during sprints, according to a master schedule.

This hybrid adaptation adds some additional meetings and discussions up front, but it also provides transparency and defines a point in the process where team members and customers can provide design input. Requirements documents can also be used by the product launch team to plan a formal release, including the development of sales collateral, marketing plans, customer service training materials and FAQs, pricing analysis, and more.

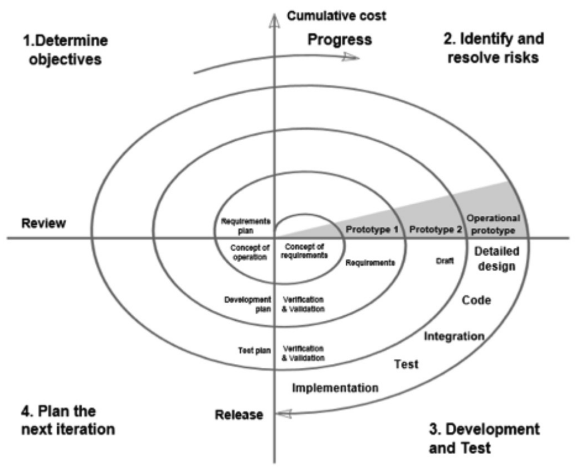
You may be wondering how the hybrid approach differs from the waterfall approach, since requirements are defined up front in both processes. The key difference is that agile provides a forum for and sets the expectation that the development team will notify the product owner and scrum master if they run into an issue that may require design changes. The design team can then make a decision in a timely fashion, and the development team can keep working. This approach encourages developers to be collaborators, not order-takers. Since development is being done in smaller chunks, this also means that if the business area needs to change requirements due to regulatory or market changes, it should cause less of a delay.

The spiral model (Figure 3) is an example of a hybrid design that follows many characteristics (prototypes, experiments, and solutions) of a pure agile development methodology. Iterations follow four key phases that are designed to identify and mitigate risks:

1. Determine the objectives and plan the scope of the increment
2. Prototyping, experimentation and research to identify and resolve potential risks (technical, conceptual, etc.)
3. Design, develop and test the increment
4. Release and monitor the increment, and use feedback to aid in planning the next iteration

Another approach is the iterative and incremental model (Figure 4). This is any combination of iterative design that attempts to address the main criticisms of the waterfall approach, since the entire project is broken down into smaller increments that apply lessons learned from previous iterations.

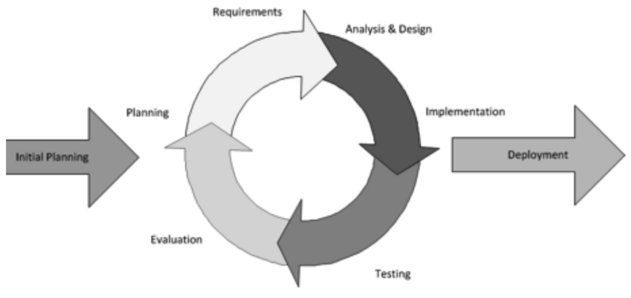
Figure 3. Diagram of Spiral Model



Source: Inflecta Corporation, Silver Spring, MD

Learning is continuous, allowing the application to evolve incrementally upon the completion of each iteration. Although this model looks very similar to agile development, there are several

Figure 4. Diagram of Iterative / Incremental Model



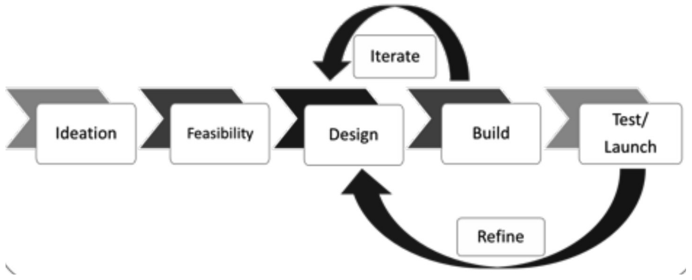
Source: Inflecta Corporation, – Silver Spring, MD key differences. Iterative development typically follows the same waterfall steps, they just occur in smaller units of time and do not have to be released upon completion of each iteration.

HOW DOES THIS APPLY TO INSURANCE PRODUCTS?

Most insurance companies are already using an agile approach in their IT areas, whereas the broader product development process is still a waterfall. Business requirements and specifications are locked down at the end of design and handed off to IT. IT then develops the technical specifications and builds the system functionality to support the business needs. In a pure waterfall approach, IT would blindly build to business specifications, without considerations of how design decisions impact difficulty or cost ("just do it"). This is suboptimal in a limited resource environment. Most companies have someone from IT participate in the design phase, but unless this person is very knowledgeable about how both the business and the system work, it can be challenging for them to be very effective. Unforeseen issues often pop up during the build phase, resulting in "rework" and slower speed to market. Unanticipated issues can also surface in compliance, actuarial, etc. While each member of the design team does their best to anticipate these issues up front, it is only natural that additional information will come out once the work is actually being done. The pressure to lock down specs and avoid rework makes the team feel like everything must be perfect coming out of design. In addition, because the entire process is so labor-intensive and takes so long from ideation to launch, sales also feels pressure to design the optimal product. This all results in analysis paralysis and reluctance to commit to decisions.

How can companies do better? One approach is to build iteration into the process (Figure 5). What looked like rework in the past now looks like ongoing design refinement. Using an iterative design and build process should reduce the pressure to hit a home run with the first set of specs. Speed to market may improve because the initial ideation through design phases are shorter, with opportunity to cycle back and make changes.

Figure 5. Diagram of Iterative Product Process



What does this look like in practice? Company X designs products by holding meetings with large groups of people (everybody wants to weigh in because they only get one chance) where design decisions such as issue ages are discussed. This process takes a long time and consumes a lot of resources.

Instead, only key decisions that affect all workstreams should be made at that stage of the process. For example, what underwriting data will be captured? Teams can then go off and work with this information.

- Actuarial can develop a mortality assumption tied to the data elements and produce pricing results.
- Underwriting can determine the best way to capture the data: application, database queries, teleunderwriting, etc.
- Compliance can draft the application.
- Systems and operations can begin their work.

As questions arise during this process, daily stand-up meetings provide the opportunity to raise questions that need input from the other areas. For example, systems and operations will need to know which databases are going to be queried so they can build in connections. The team members should be empowered to make most decisions, with the product owner responsible for deciding which issues need to be elevated to a management level (e.g., if systems identifies that working with a certain database will be much more expensive than the budget identified in the high-level cost-benefit analysis). While that decision is being made by management, other work can continue.

This is just one example. The important idea here is to only make decisions at the point in time in which they must be made in order to move forward, and to empower development teams to make most decisions. By using a collaborative and iterative process, it is less likely that showstoppers will surface late in the process without other areas being aware of the issue.

Why aren't companies using this model already? A key reason is that many of the business areas that participate in the product development process have other responsibilities as well, and their contribution to a given portion of the build may only be a day or two. Using mini sprints can help with this. Even if product resources are not dedicated, there can still be the expectation that during the mini sprint, team members are accessible and focused on their product work.

One of the challenges in using iterative development for insurance products is that it can be more difficult to do incremental releases than in the IT space. Launching a new product requires significant distribution training time. Agents can be slow to add new products to their quiver if they are already successful, and if they look at a product once and find it unsatisfactory, they might not give it a second chance. Therefore, if a company launches a product to its sales force, they want it to be something that they know will sell. This can be mitigated by soliciting agent and/or customer feedback in every step of the product process, but feedback from advisory councils and focus groups does not always capture what the actual market experience will be. This is where carriers that use direct marketing have an advantage — they can develop a product testing program that is only visible internally. This opens the door to an iterative design process that also takes into account the results of market testing. Carriers with traditional distribution also have the opportunity to pilot products in limited production. Testing can appear to lengthen speed to market since it adds an additional step to the process, but the hope is that the initial design phase is shorter since the goal is to develop a prototype, not the final product.

CONCLUSION

Agile methodology is taking the IT world by storm since it mitigates a lot of the issues that organizations experienced when using the waterfall process. That said, it is not a panacea because it can take too much design control away from the product owner. Hybrid agile approaches aim to bridge the gap. While a full-blown move to agile may not work for the broader insurance product development process, using an iterative approach can help companies avoid analysis paralysis and get to the build phase faster.

Kelly Rabin, FSA, CFA, MAAA, is a Seattle-based consulting actuary with Milliman, Inc. She can be reached at *kelly.rabin@milliman.com*.

Gary Baluta, MS, PMC, is a product management and market launch consultant based in Phoenix, AZ. Gary establishes processes and procedures that enhance the entire product development life cycle. He can be reached at *gbaluta@yahoo.com*.