
Model Exploration Whirlpool

Draft 0.3

Teams struggle to integrate modeling into an Agile or Lean software development process. The focus on completing one task at a time creates an incentive for expedient design decisions and often leads to fragmentation of both the internal design and the user experience. Over a span of time that can be as short as a few months, this can end in solutions that are complex and unmaintainable.

Avoidance of modeling and design stems from the recognition of our poor ability to foresee the future in a software project. This rightly leads to the rejection of upfront modeling and design.

Yet modeling is not the same as upfront modeling. In fact, upfront modeling is anathema to domain-driven design (DDD). That is because we view models, in part, as a distillation of knowledge of the domain in the context of our project goals. From this perspective, the worst time to do in depth modeling is at the beginning of a project, when the team knows the least. In fact, upfront modeling locks in our initial ignorance.

The Model Exploration Whirlpool provides an alternative to upfront modeling that is friendly to Agile and Lean development philosophies.

The Whirlpool is not a complete development process. It is meant to be invoked when needed within a broader process. (See section “Agile Process Hooks”.)

When to Model: Watching for Signals

In the Model Exploration Whirlpool process, modeling is not meant to be a “step” in software development; It is an activity that is undertaken at any point when the need arises.

In the Waterfall development process, modeling is a step in a project that comes after requirements analysis and before design. This has the known problems that lead to failure of complex projects. In some approaches to iterative development, modeling and development are taken in short spurts at the beginning of each iteration, which gives better results than the Waterfall, because it brings to bear knowledge learned in previous iterations, yet is still does not connect the activity of modeling directly with its benefits.

In our process, we have no fixed time for modeling at all. In fact, it is possible that a project might complete without recourse to techniques described here, if the complexity was not too great and the domain well understood by the team at the outset.

Instead, we watch for certain “signals”, indications that obstacles arising on the project are resulting from a misfit model.

Time for modeling ...

- ▶ When communications with stakeholders deteriorates.
- ▶ When solutions seem more complex than the problems.

- ▶ When velocity slows (completed work becomes a burden).
- ▶ When the team is moving into a new domain area for the first time and needs a minimal shared view to get started.

clutching a new direction for their project.

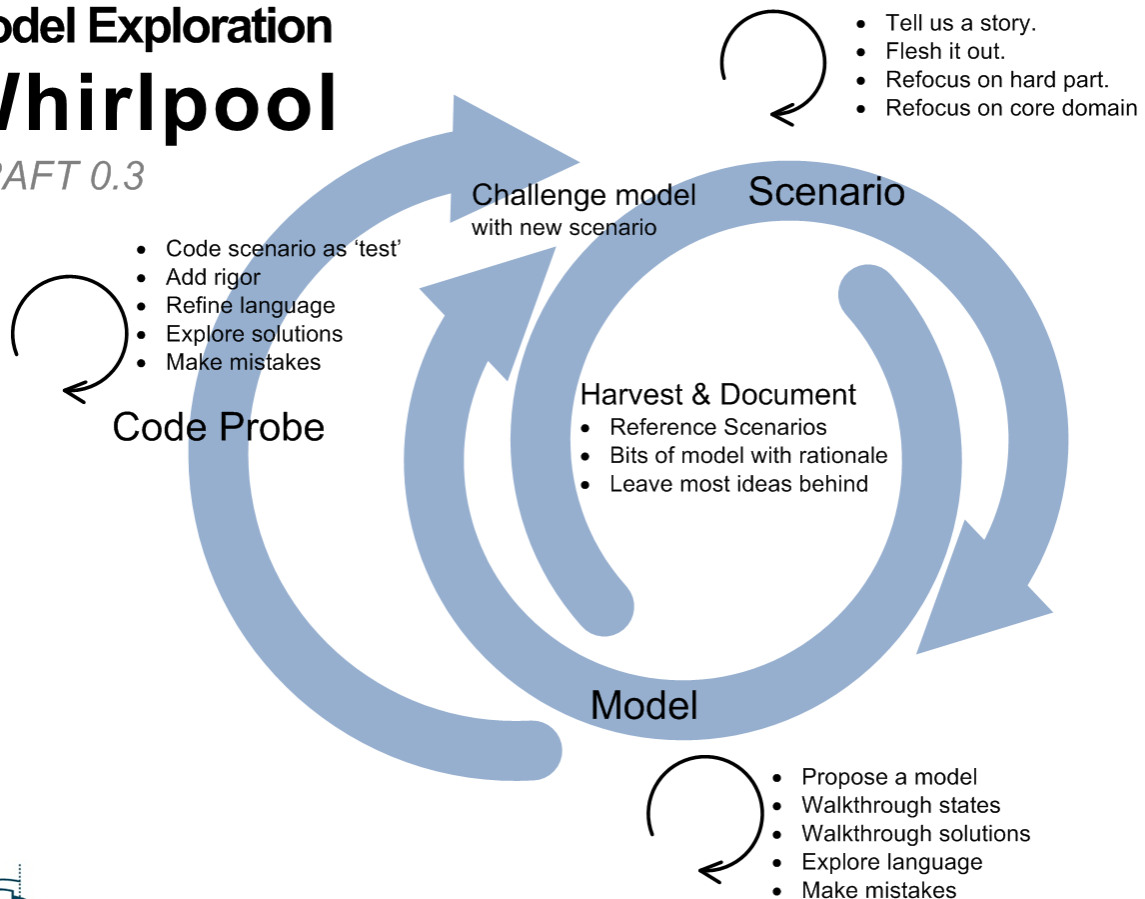
The following sections describe the activities associated with each part of the Model Exploration Whirlpool.

In the terms of Lean Software Development, we want to shift modeling from push to "pull".

When a signal is recognized, the team can "jump into the Whirlpool". They can jump in at any point in the Whirlpool; it is not a sequence. They swirl around for a while and then climb back out. Perhaps they come out having concluded that no useful modeling ideas were coming to them on that day, and they go back to their routine, to try again another day. Perhaps they come out

Model Exploration Whirlpool

DRAFT 0.3



Brainstorm

Brainstorming sessions are an integral part of both the modeling and scenario building phases of the Whirlpool. In particular, we are cultivating the creative collaboration of the technical team and their business counterparts.

The goals

- ▶ Produce or refine reference scenarios.
- ▶ Produce many alternative models of the domain.
- ▶ The brainstorm *does not commit the organization* to a development track or any other decision. It only lays out options and their consequences. Decisions about project scope are separate from these activities. This is a commitment the leaders of the brainstorm make.

Roles

Note that multiple roles may be played by the same person. The facilitator might also be a software designer. That said, a person must make clear what role he or she is playing at any given moment. Also, the same person may not play both a domain expert and a software designer.

Software designer (1+) Trained in the rigor of design and modeling. Knows the technology platform well enough to evaluate how practical a particular model will be to implement. Understands what software is capable of – able to recognize where software might help, and where it will not add value.

Domain Expert (1+) Deeply understands the domain and the concerns of the organization. Gets paid to solve problems in that domain. (This

excludes people who get paid to create software, even though they may know a great deal about the domain.) Can answer “why” questions. Not necessarily a decision maker. Does not need authority to approve scope.

Facilitator (1) Understands this process. Has the skills to keep the activity focused while allowing and stimulating creativity. Understands project well enough to judge scope of discussion (see below).

Photographer (1) Knows how to take pictures of a white-board with a digital camera.

Scope of Discussion

A modeling discussion can't be confined to the scope of the software under design, which would stifle creativity and learning and prevent the solution from considering the whole context. On the other hand, a conversation that wanders aimlessly all over the domain will not arrive at the goals.

We aim to keep the discussion within the “umbra” of the project scope. Where the boundaries are is ideally a consensus, and the facilitator should be lenient when a member of the group claims a line of inquiry is relevant, but ultimately must pull the group back into the heart of the problem.

Conversely, when the discussion gets too narrowly focused, drilling very deeply into something that is not at the heart of the problem, for example, the facilitator tries to broaden the group's investigation, possibly by asking “why” questions, or by asking the group to discuss the relationship of the part they've been examining to some other important

consideration. Again, the facilitator should be lenient, because people need to drill down in various (sometimes almost randomly selected) places to really understand, but ultimately it sometimes necessary to just declare that the group has gone deep enough into that point for now and must move on.

Separation of Decision Making from Exploration

Design idea-generation works best in groups of roughly 3-5 people, yet in many organizations, almost every meeting routinely swells because each stakeholder feels compelled to attend in case decisions are made that affect his or her concerns.

To avoid this syndrome, *fully separate the functions of meetings*, dedicating a particular meeting to either decision making or modeling/design exploration. When a meeting is designated as exploration, be faithful to the commitment to make no decisions.

An exploratory modeling and design session does not commit the organization to a development track. In some, many ideas may be generated yet none of them be deemed worth follow-up. In others, a promising idea may be analysed to varying degrees, including laying out options and consequences. This analysis provides the essential input into decision-making meetings to come.

Model exploration activities often range outside project scope (see Scope of the Discussion). However, the discussion should not be allowed to drift beyond analysis. If the time seems to have arrived for such a conversation, close the meeting

and arrange another. Invite all relevant stakeholders.

When all stakeholders trust that decisions affecting their concerns will not be made in their absence, it will be possible to have effective model exploration meetings with the right composition for those goals.

Environment and tools

- ▶ Sufficient space to sit and stand and move around.
- ▶ White board. (Flip charts as second choice.)
- ▶ Digital camera

Scenario Phase (Overlaps Modeling Phase)

Selecting the right scenarios and describing them properly is critical to successful modeling. Ultimately, a small set of carefully chosen reference scenarios will be assembled. However, it takes trial and error to finding those reference scenarios, and in this process we are exploring scenarios as much as models.

Choosing an initial scenario to explore can take from 5 minutes to 1 hour. If no clear scenario can be defined in that time, the team needs more direction about the goals of the organization before attempting this process.

Domain expert, tell us a story.

A domain expert needs to describe a sequence of events. This is not a requirements specification or a full explanation of a process. It is simply one particular path, told with particulars and motivations.

Which story?

- ▶ Something that might happen (or better yet, something that has happened).
- ▶ Something that keeps the domain expert awake at night.

It is often best to start with a quick summary of the story. Then the group can evaluate it:

- ▶ Is it in the core domain?
- ▶ Can software address (at least in part)?
- ▶ Is it in the umbra of the scope?

If it doesn't seem to fill the need, then the group works with the domain expert(s) to come up with one that does. Some leading questions can help:

- ▶ Why does this scenario worry you?
- ▶ What makes the domain problem (in the project scope) hard?

The domain experts might offer several stories before hitting upon the right sort for the session. The feedback of the group helps them zero in on the sorts of scenarios that are needed, and helps them clarify, even for themselves, the sort of help they are looking for from the software. This exploration is also part of the process, and if this takes some time it is time well spent.

Once you've found a story that is promising, the domain expert tells it in a fuller way.

Fleshing out a Scenario

First, the domain expert should just tell a story. The goal of all present should be to understand the story and understand why this story worries the domain expert.

In the process, assemble on the white board a representation of the relevant details of the story. Relevance is defined as information that affects the decisions involved in the story, or color that helps convey the realities of the situation facing a person dealing with the situation presented in the story, which help us see why it is important and difficult.

Make the example concrete and realistic.

Good Example

Bad Example

<To be inserted later>

Modeling Phase (Overlaps Scenario)

Walk through description of states in scenario

What information is relevant at each point in the story? How is this information expressed?

Keep in mind, there is not one model for a given domain, nor is there a “best” model. This is an exploration of alternative models and ubiquitous languages, and an evaluation of their clarity and usefulness with respect to a particular scenario.

As the group steps through the scenario, each participant should consciously express a description of the situation at that step in one of the proposed models, or propose an alternative and then describe the scenario step in terms of that model.

Responsibilities

Although each participant should propose model changes, practice the language, and all other activities. Even so, the people playing particular roles have greater responsibility for particular aspects.

Domain Experts Does the language make sense? Does it sound natural? Would you lose it? If you hear a phrase that sounds awkward

Software Designers Is the language precise and ambiguous? Could you program in it? If you hear ambiguities or vagueness, point it out and look for ways that can sharpen the definitions.

Facilitator Listen for differences in the language of people, their drawings, and any other communications. Are the domain and software

experts echoing each other (converging on a common language), or translating each other (staying distinct)?

Is the discussion stuck? Are participants just making minor variations around a theme? The facilitator tries to shake them loose. This is very open-ended. You might try taking away a term. “Everything seems to be focused on this one big central piece. Let’s take it out of the picture and see if we can solve the problem without it.” Your goal is not to get rid of that term, just to shake the group loose and get them generating significant variations of the model.

Photographer Each distinct idea, good or bad, try to capture whatever scratchings on the board may have been intended to communicate about the idea. Don’t be exhaustive. Just capture a reminder. No need to archive them. These pictures will be understandable for a day or so by the people who were present. If they are not used in that time (see below, Mine for Useful Bits), they never will be.

Walk through scenario solutions

<<Just find a way to merge these two sections>>

The group examines the descriptions of the state of affairs at any given moment. In this phase (which is not truly distinct from “Walking through states”, and should be loosely interleaved with it) the group is examining the transitions between those states, computations, rules and other intricacies of the domain. The goal is to find models that let you express these clearly and work through computations in a simple and natural way.

Responsibilities

Responsibilities are the same as in “Walking through states”, plus:

Facilitator Decide when the the discussion is out of gas -- No one has ideas and nothing is sparking. In this case, if the discussion has been running more than 40 minutes, do step 3 and wrap things up.

If ideas are drying up because the group has found such a neat solution that no one really wants a new idea, then it is time to Challenge the Model!

Challenge the Model

Challenge the model with a new scenario when

- You have been generating models to address one or two scenarios, you have something promising, and you want to step back for a wider view by walking through a different reference scenario.
- When you've just jumped into the Whirlpool because of incidental complexity in your solutions. Find a scenario that clearly illustrates the problems you are having. If none of the existing reference scenarios breaks the model in the same way, ask the domain experts for a new scenario. (See Scenario Phase.)

You are not looking for a quirky corner case (unless it is a case that has been causing real problems or at least worries to the organization). You are looking for another tough problem that helps us get a sense of the full project scope.

When a scenario is proposed, run through it with the favored model and see if it breaks. If the model deals with it neatly, great! Consider if this scenario is really distinct enough from the scenarios you've already examined. If really is distinct, that's an encouraging sign. Put the scenario on a list and move on.

When you have found a scenario that breaks the model and fits the criteria, go back to the Modeling Phase.

Harvest & Document

Review the scenarios and models discussed. Evaluate the scenarios for distinctiveness, relevance (how much they worry the domain experts and stakeholders), and challenge. Evaluate the models on the basis of clarity, naturalness, and usefulness in resolving the scenario challenges.

Reference Scenarios

If any scenario seemed significant, identify a person to document it. Write a paragraph capturing why the scenario is important. In a page or two, carefully represent one concrete, realistic example that was used in the exploration session.

Many scenarios will be collected. Some will be temporarily useful. Over time, some will be consolidated with others, if they are not sufficiently distinct.

Over the course of the project, one of the most valuable assets is a set of 4-10 scenarios that are carefully chosen to span the scope of the project (or a the project's work within a particular model context) and illustrate the key challenges. These "Reference Scenarios" are usually more stable than the models, and in some ways they can be more important. Every proposed model has to be able to resolve the reference scenarios.

The whole team needs to be involved in the critical task of choosing and fine-tuning these scenarios.

Model Ideas & Rationale

If any model seemed significant, identify a person to document it. Write one page or less that

describes the concept and illustrates its terminology. Capture the design rationale – why did the group think this model was worth remembering? Capture a walk-through of one or more scenarios using the model to describe the states and transitions. If there is a particular part of a particular scenario that distinguishes this model's usefulness, then just capture that part. It may be change to one relationship or interaction or one clear statement of a business rule.

Leave Most Ideas Behind

Don't worry about every idea. Much will be forgotten. Which is ok. Remember, you will often come out of a brainstorm without any new artifact. Make this be ok.

This does not have to be formal, just clear. The sketches and examples thrown up on the board during the heat of a brainstorm are often illegible. They are memory aides that can help a person who was there to recreate the essentials of the discussion if they are reviewed within a day or two. Redrawing one of these more carefully on the white board, photographing it and pasting it into a document can be fine.

This is usually a small job, unless there has been a breakthrough. What is essential is to *do it within a day or so*.

Code Probe

If a seemingly valuable idea emerged from brainstorming, you may wish to vet it and prove it out by code.

Identify the scenarios to be addressed. Confirm that the examples are concrete and fairly detailed, although you can flesh out the details during the probe.

Identify the model idea you are going to try.

Narrow the Scope & Raise the Risk

- ▶ Eliminate all CRUD
- ▶ Zero in on "What makes this hard?" You don't have to do the scenario start to finish.
- ▶ Go to the heart of what needs to be proven. Don't start safe.

Development teams are often too messy while writing production code, yet they are too tidy while exploring. Make a mess! You need to move fast. Go straight for the high risks! You need to expose them here.

Agile Process Hooks

<<Need more explanation here.>>

- ▶ Stand Up Meetings
- ▶ Spike
- ▶ Iteration Zero
- ▶ Release Planning