Von: Ward Cunningham <ward@c2.com>
Betreff: Aw: Some notes on moldablePatterns-DRAFT-2024-01-16.pdf
Datum: 19. Januar 2024 um 20:52:40 MEZ
An: Ralf Barkow <ralf.barkow@me.com>

I would like to read carefully each pattern keeping in mind that my greatest ambition would be to apply the patterns in my own environment. The authors conclude that this is believed possible. I now turn to the problem statement of each pattern as the first test of fit with my environment.

Problem. How can you cheaply and effectively extend the development environment with domain-specific tools that address your application domain?

Problem. Where do you start coding?

Problem. How do you make it easy to find interesting information again?

Problem. How can you effectively navigate between domain entities by name, content or other criteria?

Problem. How can you streamline execution of repeated actions?

Problem. Where do you start coding an explainable system?

Problem. How do you create an object in a particular state to start a moldable development task?

Problem. How do you develop custom tools for existing data?

Problem. How can you effectively provide custom tools for various kinds of collections of domain entities?

Problem. How can you keep track of decisions, experiments and progress in a moldable development project?

Problem. How do start moldable development if you are missing core infrastructure?

Problem. How do you pick the first moldable development task to focus on?

Problem. How should you design a custom view to effectively com- municate the information of interest?

Problem. How can you advance the assessment effectively?

Each statements describes how or where to apply a solution without explaining what human need causes these solutions to be routinely required. We were lead to believe that an ability to make decisions is the root of each need. I will be looking for ways to understand this decision making better and to tie human decision making to the solutions offered by each pattern.

On Jan 19, 2024, at 11:42 AM, Ward Cunningham <ward@c2.com> wrote:

Some notes on moldablePatterns-DRAFT-2024-01-16.pdf

I have been encouraged to read the executive summary of a business plan sentence by sentence looking for assumptions not addressed before the summary is over. I will apply this technique recognizing that the authors are not asking for investment of the same kind.

Moldable development supports decision-making by making software systems explainable.

“Moldable” is a type of development, itself a process that requires decision-making by someone, who?
A system made “explainable” provides better support for decision-making, but how? And how much?

Jason Fried writes, "It's all a judgment call. Every human decision is a gut decision. The decision itself is the messy integration of many disparate pieces of information, experiences, instincts, stories, and unknowns.” https://world.hey.com/jason/it-s-all-a-judgment-call-6ab5d025

This is done by making it cheap to add numerous custom tools to your software, turning it into a live, explorable domain model.

Tools make software explorable by navigating a model, but is the model reliable? Sufficiently complete?
The model will be in terms of the problem domain, but does the domain have ubiquitous terminology?
The model will be live, meaning it will acquire experience like any living thing?
The tools will be cheap, but in what way? Inexpensive? Prone to breakage? Of no lasting value?

Based on several years of experience of applying moldable development to both open-source and industrial systems, we have identified several mutually supporting patterns to explain how moldable development works in practice.

The authors base this publication on experience, direct personal experience in each case? Reports from others?
Several repeating solutions have been identified and described as patterns using words and pictures.
The patterns depend on each other in order to provide solutions, presumably leading to explanations.
The patterns explain the development process, but not the system that is explained by the process leading to decisions.

I now leap to the conclusion and continue careful reading. I can't be critical of omission now since these sentences complete the document, rather than introduce it.

Moldable development makes software systems explainable by making it cheap to add custom tools that support decision making.

Recapitulation of above.

In practice, however, requires a paradigm shift since it leverages live programming to explore domain objects and incrementally mold them.

Live programming is a paradigm shift.
The shift is required to explore.
The shift allows one to mold the model as explored.
The molding is done in increments.

The pattern language described here has been mined from several years of experience in applying moldable development to both open- and closed-source projects, using the GT moldable development platform.

Recapitulation of above.

Although the examples given here have all been drawn from GT, we believe that they can be applied in any programming environment that offers a minimal level of support for live programming.

GT embodies these patterns.
The authors believe the patterns have utility not tied to GT, but are they sufficient to match GT when applied?
To be useful the patterns require some aspects of live programming, which aspects?

This last sentence is sufficient reason for the makers of other environments to pay attention to this work.The claim is qualified as only a belief of the authors. Surely these patterns will prove valuable should the authors of GT undertake GT 2.0 and want to be sure not to lose something important. I am left wondering what sort of project should be undertaken in my own environment and how that project's result should be evaluated.