EASTERN WIND
罡风咨询顾问有限公司

根据话题的洞见	Industry Insights
基于模型的敏捷方法论	Model-based Agile
2019 年 03 月 31 号	March 31st, 2019

Introduction

Ever since the first software program went production, developers have been seeking methodologies to better manage uncertainty and complexity in the face of rapidly changing technologies and business models:

Structured Programming

Structured Systems Analysis and Design

Object-oriented Programming

Rapid Application Development

Rational Unified Process

Extreme Programming

Agile Unified Process

Scaled Agile Framework (SAFe)

Large-Scale SCRUM (LeSS)

With all the benefits that Agile-based methodologies offer, we've observed that, as a human endeavor, software design still has room for improvement.

Not So Agile…

The underlying premise of Agile is that "knowledge comes from experience and making decisions based on what is known".

Agile methodologies use an empirical process of transparency, inspection and adaptation to make decisions based on knowledge uncovered in each iteration. In theory, Agile:

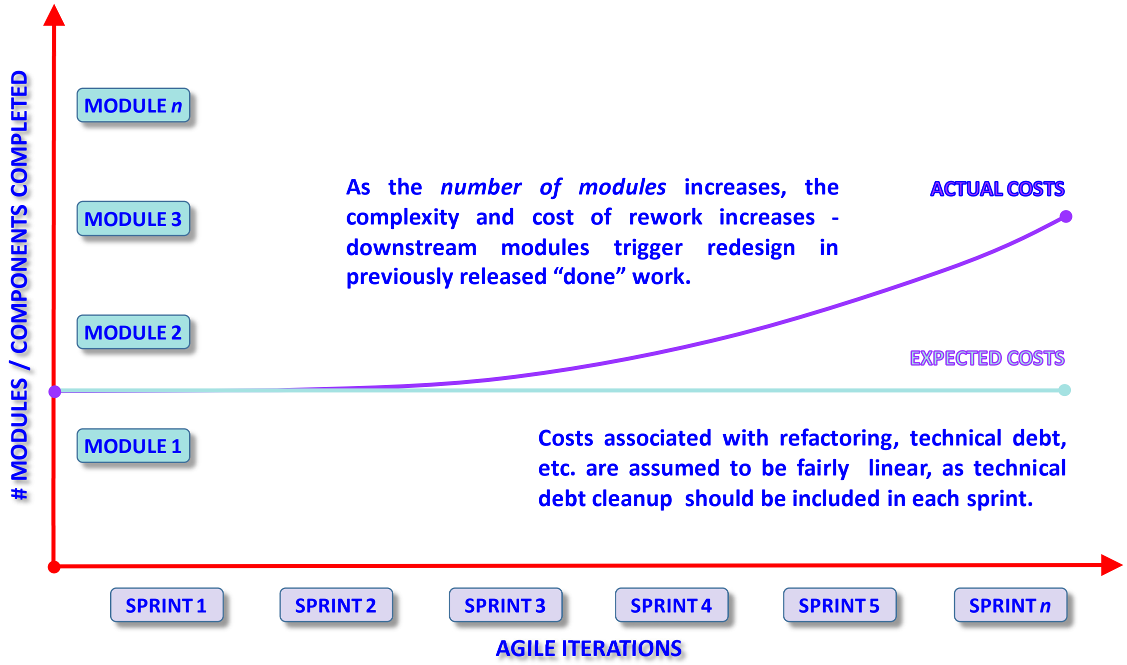
- Delivers business value sooner than traditional approaches.
- Is adaptive, especially in the context of a rapidly changing business environment.
- Provides greater transparency to all stakeholders, as tangible product is delivered in regular increments.

- Reduces risk by encouraging integration sooner rather than later.

But in practice, we've encountered projects that don't conform to expected outcomes…

- The software components developed in later stages of an Agile-driven project can have a material impact on components that have already been produced and released, thereby triggering redesign of "done" work.
- As a project progresses, the cost of resolving technical debt associated with late stage component development is cumulative – as a project's portfolio of implemented functionality increases, so does the total cost of any rework.
- The prioritization algorithm for managing backlog items may not always result in a scalable or extensible application architecture.
- Finally, since teams are (presently) made up of humans, there is a tendency to focus on simpler use cases at the outset of a project – more complex use cases tend to have more dependencies on other components, some of which may not have been developed yet.

How can we address these issues without compromising the real value of Agile methodologies?



As the *number of modules* increases, the complexity and cost of rework increases - downstream modules trigger redesign in previously released "done" work.

Costs associated with refactoring, technical debt, etc. are assumed to be fairly linear, as technical debt cleanup should be included in each sprint.

Borrowing from the Building Trades…

In the construction sector, there are established (and self-evident) procedures for building a house.

No carpenter in his/her right mind would dream of building a house without a blueprint, a foundation and a frame. The thought of building rooms in accordance with which one has the "highest value" doesn't make sense – one wouldn't start construction by first building the kitchen, then the upstairs bathroom and then the living room (if those happened to be your priorities).

In many respects, developing software is analogous to building a house – you need a blueprint, a foundation and a frame, in that order. But software diverges from this paradigm in the dimension of innovation – regardless of the particulars of a house design, it's generally a variation on a time-tested model. Software, on the other hand, is constantly evolving at a rapid pace.

Is it even possible to create a blueprint, foundation and framework model for new software development? We think the answer is yes…and fairly recent developments in enabling technologies make a case for an optimized Agile methodology that can consistently and predictably manage complexity.

Model-based Agile

We call the approach we've developed "model-based Agile" (MBA) – where process models are the "blueprint", data models are the "foundation" and prototypes are the "frame".

Some might argue that any Agile approach that uses process and data models is a back-handed way of reverting to traditional SDLC methodologies. But the premise of this approach is that Agile-based projects can benefit from an integrated model without compromising core Agile principles.

The underlying foundation of MBA is the use of process and date models that:

- Provide a holistic context for each sprint and each Agile team, so that developers can focus on selected backlog items knowing that "done" work will generally support downstream components.
- Initially focus on breadth vs. depth, in the sense that they represent the end-to-end business model to be supported by the project.
- Provide only as much detail as is necessary to support each sprint – and, in fact, are elaborated in parallel with each sprint.

BENEFITS

In our experience, MBA can substantially reduce risk, minimize rework and enable teams to deliver products with a significantly lower total cost of ownership (TCO) – at its core, MBA is a TCO-driven methodology:

- Backlog items are organized by process group, so that the entirety of a process is known, even if a subset of a process group's associated backlog is implemented in any given sprint.
- Backlog item selection is based (in part) on the foundational value of a component, meaning that backlog grooming and sprint planning fully incorporate dependencies between backlog items.
- Risk and complexity can be identified up-front so that costly design errors can be avoided.
- Risk and complexity management is based on progressive elaboration – it doesn't force teams into long-running (and unproductive) analysis.

ORGANIZATIONAL IMPACT

A key principle of MBA is that it must support, not disrupt, Agile teams. To that end, we'd like to propose the role of the "MBA Architect" - essentially a business architect who can serve up near-term process and data requirements in the context of a holistic model.

The MBA Architect acts as consultant to the Product Owner, ensuring that the backlog is aligned with a future state business model.

The MBA Architect also acts as a consultant to the development team, guiding them in their design efforts by highlighting downstream process and data issues that impact the design of any given sprint.

The MBA Architect must be highly sensitive to the cadence of a sprint by introducing relevant knowledge of downstream processes and data as a constraint to the sprint.

On smaller projects, an MBA Architect may participate as both a consultant and an analyst on the project team (although we'd like to note that a business architect and a business analyst have distinctly different skill sets).

ENABLING TECHNOLOGIES – BPM

Until relatively recently, analysts lacked efficient tools to document business processes. The chief limitation in using simple diagramming tools (e.g. Visio) is a lack of support for process decomposition. The inevitable result is that processes don't enable stakeholders and developers to focus on the right level of detail in each sprint – process discussions become a distraction, rather than an enabling mechanism.

A secondary limitation of diagramming tools is the inability to integrate data, functionality and other requirements in a holistic dependency analysis. As a result, Visio-based process models are one-off artifacts, distinct from the rest of the development effort.

In the last 10 years (give or take), process modeling repositories have matured into fully integrated requirements modeling tools that enable business architects to work in a top-down fashion:

- Process decomposition enables the MBA Architect to drive discussions with the Product Owner that simplify processes and identify root-cause impediments to optimized future state processes.
- At the same time, process decomposition lets development teams focus on precisely what is necessary to complete a sprint, while the MBA Architect continues to elaborate the process models ahead of the next iteration(s).
- Integrated repositories' sophisticated dependency analysis help identify risks before designs are coded.
- Finally, repository-based tools are focused on associations and attributes, not text – in keeping with Agile principles, requirements are terse.

Signavio, a Gartner's Magic Quadrant® leader in the BPM space, is a cloud-based BPM solution with an intuitive, yet powerful, interface for progressive process decomposition and active metric-based process optimization.

Enabling Technologies – Data Modeling and Prototyping

Low-code application development platforms have finally reached a level of maturity that they can now be used by development teams for data design, wireframes, prototyping and, in some cases, as the primary app dev platform.

Mendix is a Gartner Magic Quadrant® leader in the low-code aPaaS space – it is an integrated object-oriented data modeling tool capable of generating fully-functional apps that can be enhanced with a "non-programmatic" microflow rules engine. We've used it successfully to develop fully working applications in a fraction of the time it would take to write custom code.

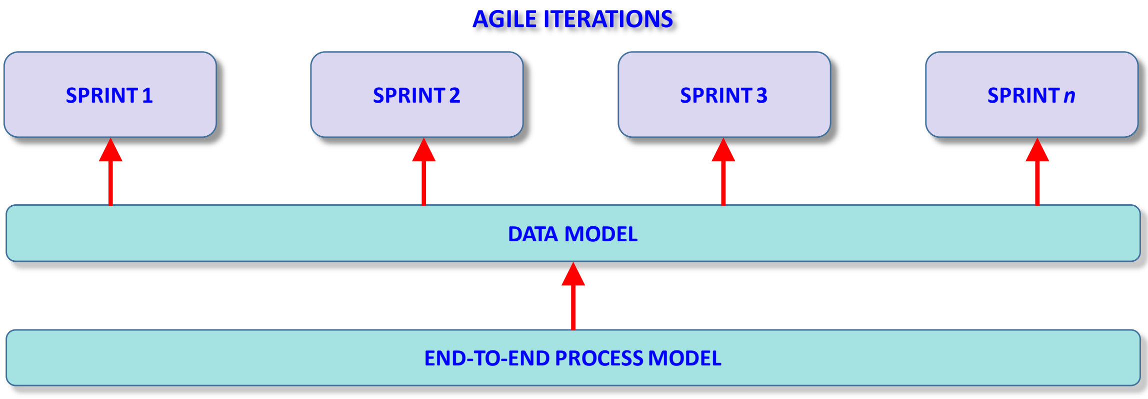
But more often than not, we use Mendix as the primary data model, wireframe and prototyping tool in our MBA toolset:

- Object-oriented data design makes it possible for the MBA Architect to design a complete high-level data model to support all end-to-end processes.
- Generalizations can then be decomposed into specialized classes as the project proceeds through each sprint.

- Standalone wireframes are no longer necessary, as working screen and report prototypes can be quickly generated from the data design.

MBA Methodology Architecture

MBA artifacts are the foundation for each successive sprint – the methodology stack looks something like this:

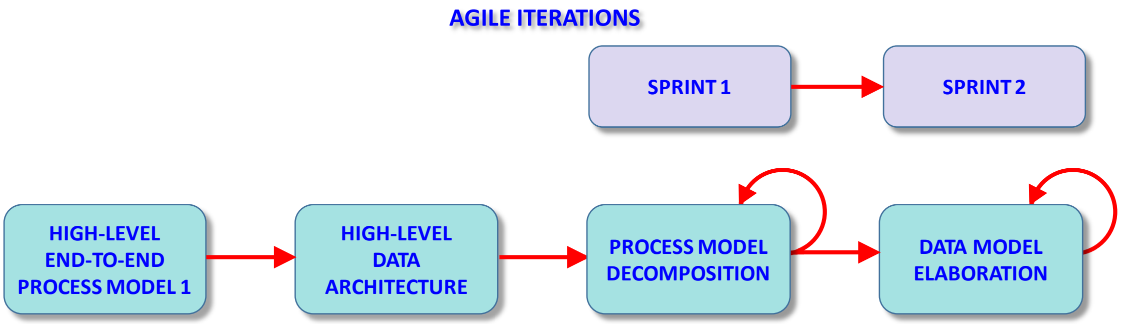


MODEL-BASED AGILE ARCHITECTURE

Implementation Considerations

In keeping with our mantra of low-impact on Agile teams, we recommend that the MBA tasks and deliverables be loosely-coupled with Agile product development.

Notwithstanding this loosely-coupled relationship, MBA introduces a certain lead-time issue in order to prepare baseline process and data models to be used by the development team – the MBA Architect needs the equivalent of two “sprints” to prepare the initial models.


AGILE ITERATIONS
SPRINT 1
SPRINT 2
HIGH-LEVEL END-TO-END PROCESS MODEL 1
HIGH-LEVEL DATA ARCHITECTURE
PROCESS MODEL DECOMPOSITION
DATA MODEL ELABORATION

MODEL DEVELOPMENT ITERATIONS

Conclusion

As with every methodology that has preceded it, Agile continues to evolve. And while SAFe and LeSS address issues related to large-scale projects, we think MBA addresses issues specific to long-running projects with complex processes and data requirements.

There is no perfect once-size-fits-all approach to managing complexity. But we've seen significant risk and cost reduction when high-level process and data models inform better design decisions across all sprints.

About Eastern Wind

Eastern Wind is a digital consultancy based in Shanghai, PRC. We provide local Chinese and multi-national companies with digital strategy, program management, CRM and digital marketing systems design and integration services.