Legacy modernization

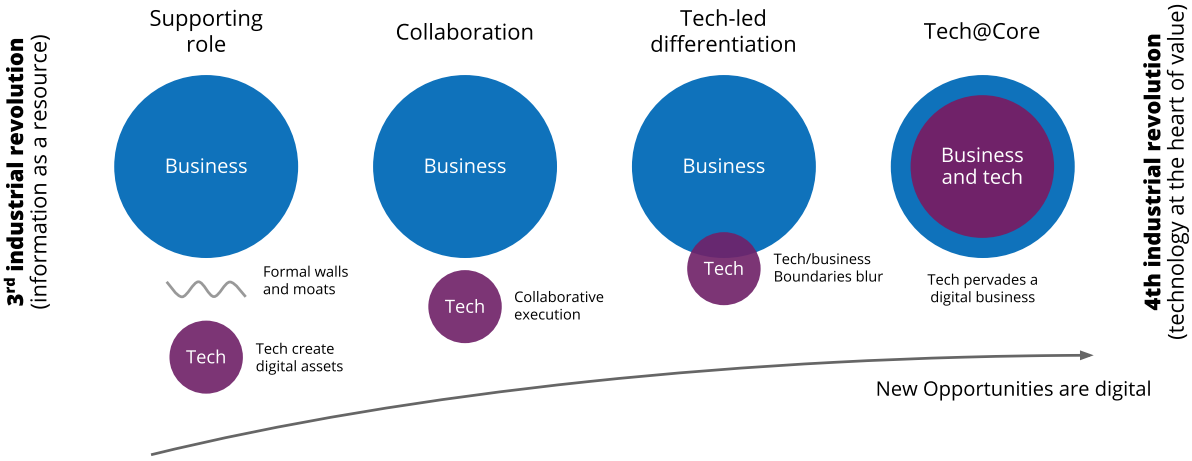
An opportunity to reimagine your future business

ThoughtWorks®

The world around us is changing epidemically. Just turn back a few months of disruptions that were forced upon organizations - few never estimated while fewer anticipated return to 'normal'. These changes would soon be classified as post 'new-normals' awaiting a future that is beyond imagination.

Nimble organizations — those with technology at their core — have the advantage over enterprises hindered by legacy systems, organizations and mindsets. The Fourth Industrial Revolution creates a deeply integrated relationship between technology and business.

Evolving relationship between business & technology



"What we need to do is always lean into the future; when the world changes around you and when it changes against you - what used to be a tail wind is now a head wind - you have to lean into that and figure out what to do because complaining isn't a strategy."- **Founder, CEO Amazon, Jeff Bezos**

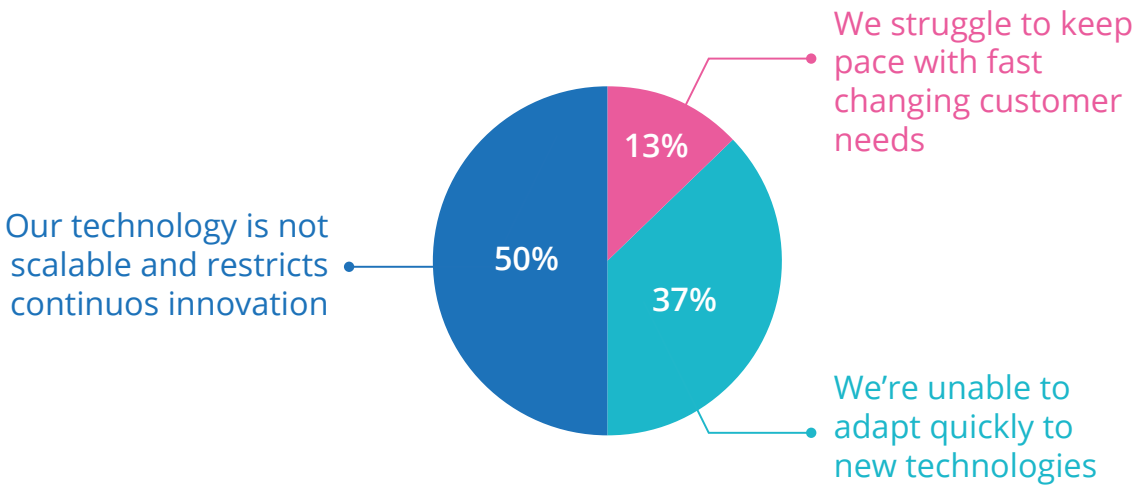
As leaders in the organization, this is the time to be bold about what happens next. You need to reframe your various perceptions of 'legacy modernization' and stop considering it in silos of 'technology', 'debts' or 'replacement,' but rather, reimagine your future business.

The demand for Business Agility

There could be multiple drivers in the organization that would demand for business agility with ambitious goals.

Continual\Sustainable change *"Business can only move as fast IT can deliver"*	Innovations *"Stay Competitve"*	Organizational Transformation *"Technology at core"*	Technology Leverage *"Competitive Advantage"*
▪ Reduce time to market ▪ Frequent Frictionless Releases	▪ Quick experiments with fast feedback (without disturbing unrelated systems and processes) ▪ Data informed decisions	▪ Business-tech partnership ▪ Embrace change ▪ Close collaboration over hierarchical ▪ Culture of learning ▪ Talent retention and attraction	▪ Increase efficiency and productivity (eventual cost reduction) ▪ Mitigate Against EOL stack (end-of-life) ▪ Improve Auditing and monitoring (e.g. SOX) ▪ Simplify PCI compliance ▪ Improve security

In a ThoughtWorks Live survey of 72 senior executives from across 42 leading global organizations, 65% agreed that their technology is holding them back. Of these, 50% described the problem as an inability to scale and enable continuous innovation, 37% said they were unable to adapt quickly to new technologies, and 13% said that they struggle to keep pace with fast-changing customer needs.



If you don’t understand legacy, you can’t achieve agility

The cool tech of yesterday is the legacy of tomorrow. Legacy modernization is the most important investment a business can make in future innovation and competitiveness – and it involves far more than building a new IT system. Legacy infrastructure, legacy architecture, legacy code and legacy data contributes to stifle business agility and innovation. So what do we mean by legacy?

Legacy Architecture: Unable to respond to the business needs of the present. Sometimes it is also too expensive to maintain (Opex) e.g. Mainframes.

Legacy Code: Too complicated, fragile, rigid and difficult to maintain or scale. Missing test coverage.

Legacy Infrastructure: Hard to scale, fragile, difficult to troubleshoot, hard to maintain.

Legacy Data: Too many data debts and hacks, don’t give much valuable information, inconsistent, not trustworthy.

Legacy Process: Tedious process with human to human handoffs, difficult to track and easy to make mistakes, difficult to sustain, transition, measure, etc.

Legacy Thinking: Thinking that all of the above cannot be changed, not agile, follow hierarchy, not aligned, Limited understanding of problems, Biased opinions.

For any organization, strategic and business critical systems always span across all of the above.

Legacy modernization is a continuous process of improving these strategic and business critical systems with a goal of achieving business agility.

Mental Models - The elephant in the room

Breaking out of legacy for business advantage is hard. There are many known barriers to overcome, starting from real system complexities and articulating business value, to investment constraints, etc. There is one very important hidden barrier that makes sustaining modernization more difficult - the disbelief and perception (**mental models**), held by individual contributors and leaders in the organization, about the overall problem space and the goal. To understand better let's try to answer the following question:

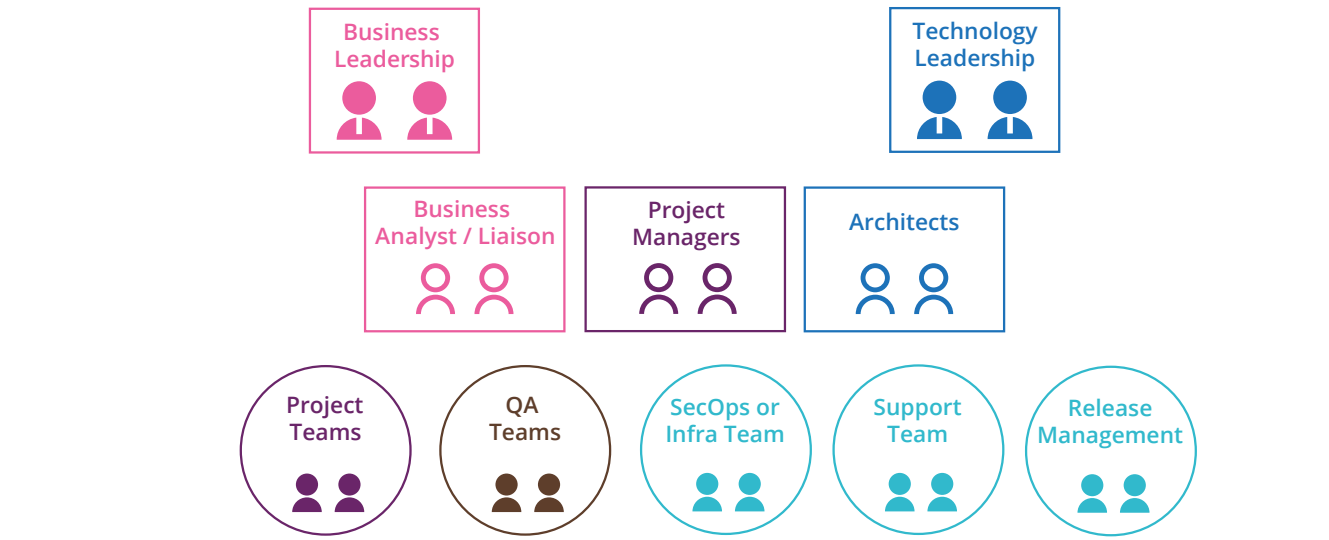
Q) What would you do if your goal is to make money?

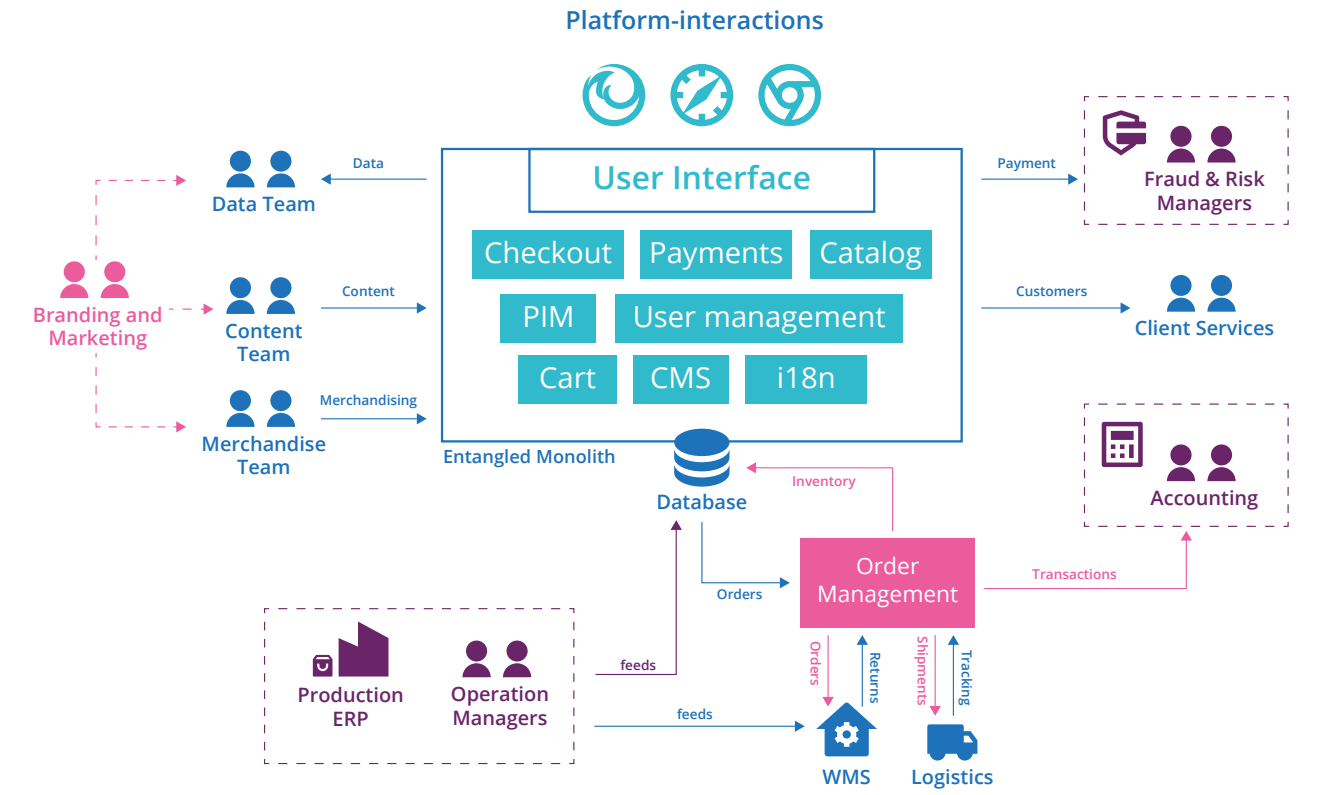
- Some may take the less risky approach with consistent predictions e.g. work extra hours
- Some may believe in taking high risk with a possibility of making more money e.g. stocks, gambling
- Some may just readjust the goal and equate more savings to having more money
- Some may want to argue on the goal itself and perhaps don't want to proceed without agreeing as to why they should make more money

The point here is that our mental model sees the information and processes it in a very different way which could be influenced by emotions, positive attitude, facts, past failures, perceptions, etc. These differing mental models in an organization with various interpretations of goals, definition of problems, and approach towards the solution is what makes it harder to sustain progress in modernization.

Let's consider another example of a traditional hierarchical organization with an entangled e-commerce platform. This type of organization typically consists of various siloed project teams responsible to enhance and maintain the platform. These teams get requirements from intermediaries (business liaison) and would work with other shared teams such as QA, Security, Infrastructure and Release Management. There are other teams/departments within the organization that would interact with the platform such as Data Team, Accounting, Client Service, etc. Each of them have their own perspective and lens to define the constraints and challenges. This results in differing beliefs and perceptions about how they can contribute to the overall goal.

Traditional hierarchical organization





Technology Leadership

- *"It was all entangled monolith before I joined the organization"*
- *"I have many ideas regarding cloud adoption, data strategy, microservices etc. We just need better talent in the organization"*
- *"It took us years to mature the business logic in current legacy stack. We should now just focus on performance and scale while reusing the core business logic as-is"*

Business Leadership

- *"We have unique ideas but it takes weeks or months to put somthing into production"*
- *"Our next 5 years strategy is clear but the technical organization is very slow in responding to any changes"*
- *"We need a platform where we can experiment our great ideas and stay competitive"*

Project Teams

- *"It takes really long to setup our workspace, we should just dockerize everything"*
- *"We are daily busy in random organizational mandatory and planning meetings, we have no time to code"*
- *"We always wanted to do cool stuff but we never had management support"*
- *"The business kept on changing the requirements, we re-wrote the code multiple times hence there was no time to write automated tests"*
- *"We cant follow any good technical practices because the timelines are always unrealistic"*

QA Team

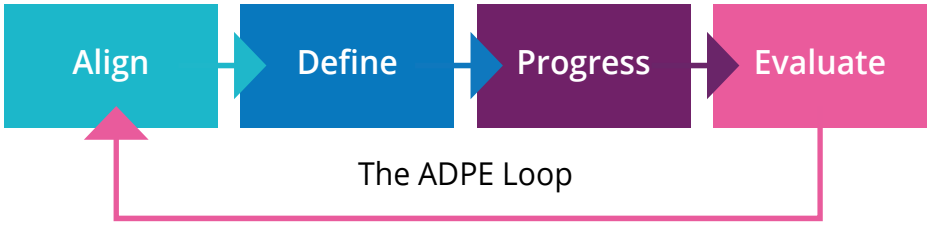
- *"We would like to write automated test, but we are always busy with manual testing"*
- *"We are never involved during the development phase"*
- *"The current application is too slow which makes the automated test slow"*
- *"The development team always break our automation suite as they add feature or refactor code"*
- *"Defect age and turn around time is too large, there are multiple teams working on the same codebase and it's hard to identify who broke what"*
- *"The development team always leaves the environment unstable by deploying new features / incomplete code without communication. This impacts our productivity"*

Data Team

- *"We are always busy in fixing the current data warehouse inconsistencies"*
- *"We are never engaged from the beginning of the project/feature. Businesses always ask for reports when the feature is already in production"*
- *We should have invested in Data appliances instead of current home grown warehouse and ETL-everywhere solution*

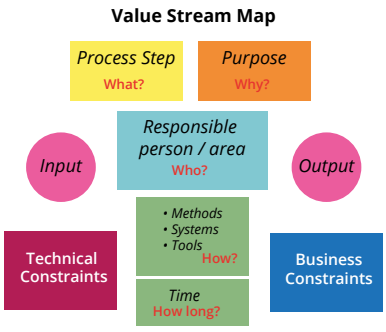
A holistic approach to legacy modernization

By taking a holistic view of modernization, organizations can create the opportunity for sensible and effective change. It consists of four key stages that enable business and tech to align, develop a realistic strategy and vision with technology aligned to the real driver of value, make progress in smaller increments with a responsive organization, and finally evaluate, learn and repeat. We call this the **ADPE** (Align, Define, Progress and Evaluate) **Loop**.



Step 1: Business and Tech Alignment

Business and technology alignment is an extremely important first step towards modernization. The goal with such alignment is to look at the drivers or motivations behind change and the goals (desired state) **as a group**. At the same time it is equally important to align our mental models on the way we look at problems by understanding existing **Technical** and **Business** constraints.



One of the techniques could be to visualise **Path to Production** or more precisely **Value Stream Mapping** tailored for your specific context (e.g. focusing on the path-to-production with respect to tech stack, maturity of your dev-ops and release process, handover between teams, etc.). The important factor here is to align as an organization on the **current constraints and our desired state**.

Here are a few examples of constraints and desired state that could be depicted in such alignment:

Technical Constraints

- On prem infrastructure
- Dependencies on infra team
- Proprietary stack with longer learning curve
- No automated tests (or extremely slow)
- No logging and monitoring
- Inconsistent and duplicate data
- Entangled monolith, huge codebase
- Too many environments to manage
- Unreliable reference data or scattered source of truth

Business Constraints

- Time to market
- External / Internal deadlines
- Budget
- Regulations
- Organization structure
- Competing priorities between different business areas
- Data inconsistencies between processes

Desired improvements

- Decoupled cohesive services
- Modern programming language
- Pipeline-as-code, Infra-as-code
- Automated test with quality gates
- Moving to cloud e.g. AWS, Azure, GCP, AliCloud
- Self service data e.g. allowing business to query data and generate reports
- DevOps and SecOps within teams
- User behaviour tracking, A/B testing

Step 2: Define

The next step is to define a technical vision and strategy (here the assumption is that you already have a business vision and strategy defined). While depicting the technical vision and strategy, It is important to consider all information at hand with respect to business vision, goals, priorities, existing pain points, business and tech constraints, and desired state.

The Seven Commandments

A secret sauce for a good technical vision and strategy is to consider these seven commandments, based on our experience at ThoughtWorks.

"Thou shalt factor in enough flexibility"
It's important to factor in enough flexibility in tech strategy for responding to business changes (vision, priorities, etc.).

"Thou shalt not waste time in discussing and choosing a single approach up front"
A common misconception while discussing legacy modernization is to identify and adopt one of the approaches i.e. reWrite vs reFactor vs rePlatform vs reArchitect. There is more to this confusion on deciding between buy vs build. Understand that most enterprises will eventually do all of the above. As you work out the tech strategy, you would define which parts to reFactor, which to rePlace with off-the-shelf, and some parts you may end up reArchitecting.

"Thou shalt always respect value driven increments"
It's important to have a broader vision with an immediate focus on small value driven increments - also try not to make it a "play with new technology" initiative.

"Thou shalt not disrupt critical business areas"
Critical business areas must continue to function and respond to change throughout the modernization process.

"Thou shalt always keep it transparent"
It's important to stay transparent with the development teams on the opportunities in the near-term target start of the platform. At the same time, you should also stay transparent with

the business about the additional cost and complexities as the platform goes into a transition state.

“Thou shalt respect the law of diminishing returns”
Be pragmatic while defining solutions. Try to consider the 80-20 rule (Pareto Principle) i.e. 80% of outcomes would come from 20% of inputs or causes or efforts. Avoid over engineering or achieving perfection.

“Thou shalt not create entire target state architecture up front”
It is not appropriate to create a full target architecture blueprint since you will want to experiment and factor in lessons from the transition state, changing business priorities, changing tech landscape, etc.

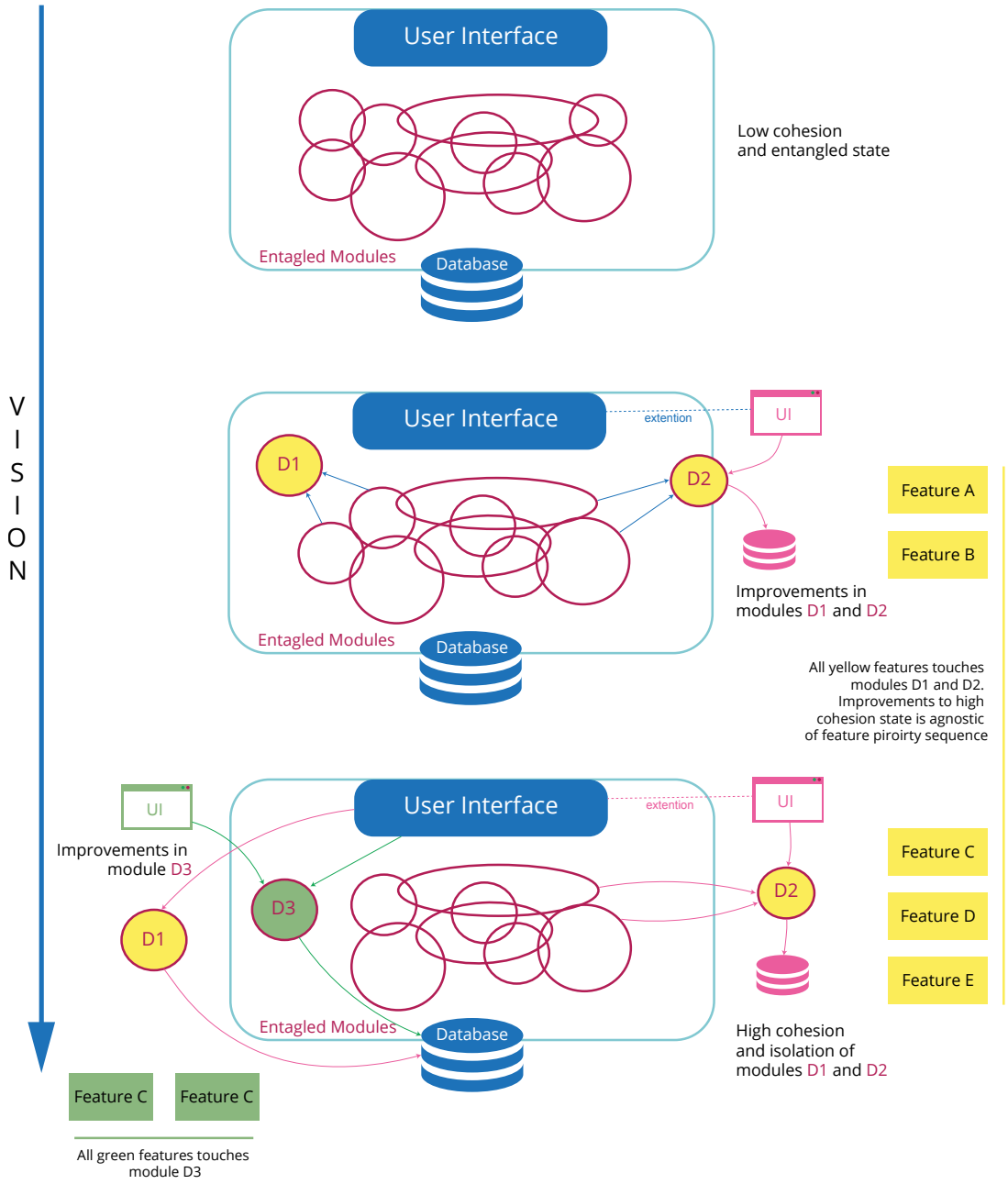
Transition State

As the architecture evolves, consider learning from various transition stages as part of your tech strategy. It is important to identify some key short-term architectural bets. The transition state would help in validating these bets with a potential to progress in long-term adoptions. Some examples could be:

- Given your current entangled monolith where your desired state is to have an API platform, it would be appropriate to start modularising it first and learn, and then slowly achieve the state of cohesion.
- Data ownership within teams (short-term) to Data Mesh and Self Service Data (long-term).
- CI/CD, pipeline-as-code for new features (short-term) to infra-as-code and elastic infrastructure (long-term).

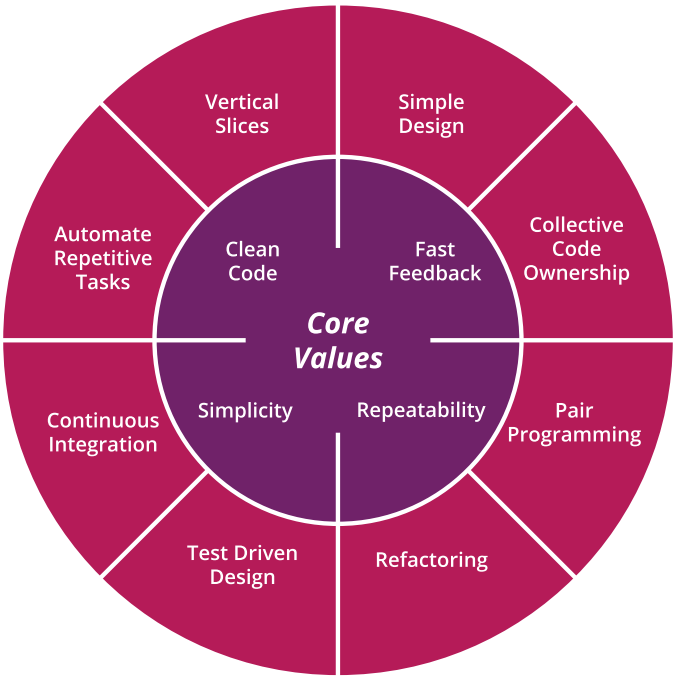
Vision Map

Create an architecture vision map to depict the progression for short-term to transition state to long-term improvements mapped to prospect business priorities with no guarantee of a particular order. For example, in the figure below: the modularity/improvements in 'X' key domain areas can be initiated by any of the features in yellow (that touches 'X') irrespective of which feature is prioritized first and then the next feature in yellow would help transition it to next stage of maturity i.e. high cohesion and isolation of those modules.



Software engineering - Core Values and Practices

Adopting the core values and practices of software engineering helps in building software that will not only delight users and customers, but will also allow for rapid change with confidence, both in short and long term. Below are four core values and eight practices that we strongly believe in ThoughtWorks - some of which are adopted directly from XP values and practices.



Step 3: Progress

As you start your journey, you have to spend more time initially in converging on implementation strategy until you reach a more mature self-sustainable state. Remember the end-goal of legacy modernization is a responsive organization with the agility to keep evolving to meet emerging customer needs. It is certainly not to reset the clock on legacy issues and buy another decade or so before time catches up with the business again. Some contextual questions could be as follows:

- Should we start with one small team or multiple?
- Do we need to integrate more specialized people in the pioneering team?

- Do we need some external help to implement the strategy or make progress?
- Can we have appropriate talent and mindset within teams?
- How will we take care of the BAU backlog?

Step 4: Evaluate and Repeat

It's important to measure if you are really making progress towards improvements and desired state. A holistic modernization scorecard can help you understand and evaluate the fitness of your organization. Examples might include the DevOps four key metrics such as release times, mean time to recovery, and incident response - but also robust indicators that are less metrics than business benefits such as the speed of change, ability to innovate or employee retention.

Capturing engineering metrics and having a real-time view of DevOps health is equally crucial. There are many solutions that you could consider based on your ecosystem such as gocd: Actionable Continuous delivery Metrics, DevOps Macro view, Azure Devops, etc.

Don't just update systems, transform organizations: The most successful cases of legacy modernization are not stories about a system updated but an organization transformed. Modernized businesses talk about greater efficiency and dramatically reduced time to market for new initiatives. But they also talk of a transformed relationship to innovation – and a transformed relationship with newly energized employees.

It's also important to realize that modernization can be a difficult undertaking specifically for the tech landscape. There is no silver bullet to give you long term answers for the choices you could make now. It's a continuous process of Align - Define - Progress - Evaluate with a goal to provide business agility by improving strategic and business critical systems!

Author

Nouman Memon
Lead Consultant at ThoughtWorks

About ThoughtWorks

ThoughtWorks has spent its last 26 years partnering with organizations to modernize their technology in order to meet customer's demands, remain competitive and grow. During our work with the transformed organizations, we've identified a common set of characteristics that they all need to exhibit, the principles critical for successful modernization/transformation and how to build a modernization roadmap using these Agile principles to drive and scale enterprise change.

CONTACT US

Get in touch with us at
contact-us@thoughtworks.com

ThoughtWorks®

thoughtworks.com