

STRATEGY ALIGNMENT

1.

- Any alignment with strategy is coincidental or opportunistic.

2.

- Upfront engagement with stakeholders to ensure business and technical alignment.
- The product/project vision is explicitly aligned to strategy
- Post-implementation review to confirm strategy alignment.

3.

- Occasional engagement with stakeholders throughout delivery cycle to review business and technical alignment.
- Backlog items are created to deal with strategy alignment issues.

4.

- Frequent engagement with stakeholders to review business and technical alignment.
- Team presents product to architecture and design teams with the intent of picking up alignment issues.

5.

- All major strategy alignment backlog items have been resolved.
- Metrics to measure strategy elements defined and tracked.
- All requirements are attached to a business metric.
- Metrics are tracked over time and improvement targets set and achieved.
- Team independently innovates by creating new features or optimisations in support of or extending the strategy.

PLANNING AND REQUIREMENTS

1.

- Team is not involved in estimates.
- No specific prioritisation of requirements.
- Iteration lengths are erratic based on the amount of functionality
- The team does not know their velocity.

2.

- Team performs estimates up-front.
- Requirements are prioritised based on business value.
- Iteration lengths are fixed.
- The team knows their velocity.

3.

- When the team estimates the estimates include all activities to reach 'Done'.
- Team performs estimations iteratively.
- The MVP and MVFS have been defined.
- The team velocity is predictable.
- All backlog items are sized by the team.
- Stakeholders attend showcases.

4.

- Team tracks performance against estimates.
- Requirements are developed on a just-in-time basis.
- Analytics are implemented to determine the effectiveness of requirements.
- The amount of functionality for each iteration is determined by the team's velocity.
- Technical Debt and Defects are tracked on the backlog and form part of the estimated team velocity
- Stakeholders actively participate in retrospective

5.

- Requirements are defined with an expected outcome with an objective measure.
- Release planning is performed based on the team's current velocity.
- A process is in place (e.g. a formal beta program) to allow extended stakeholders and customers to evaluate the software and provide feedback.

CODING PRACTICES

1.

- No standards or mechanism for ensuring code quality

2.

- Guidelines and/or standards are defined.
- Consistent training process for new team members is in place.

3.

- There are mechanisms in place to ensure that standards are followed.
- Process and practices are understood and followed by all.
- Metrics are defined but not necessarily reviewed and acted on.
- The team proactively improves the code.

4.

- Code Metrics are part of build automation and continuous integration.
- Code Metrics are tracked for trends and adjustments made on a continuous basis.
- The team regularly performs katas with the objective of improving their skills.
- Code is regularly refactored as part of the iteration.

5.

- Standards are regularly reviewed and updated.
- Processes and Practices are regularly reviewed and updated.
- The code metrics are assessed and backlog items created to drive improvement.
- Technical Debt is minimised.

CONTINUOUS INTEGRATION

1.

- No version management of artefacts and reports.
- Deployments & Rollbacks are manual.
- Build is performed manually and infrequently.
- Build is owned by a specific person.
- Testing is manual

2.

- Source code is under version management.
- Build is automated and tests are run as part of the build.
- Testing partially automated, and code coverage greater than 0

3.

- Team checks code into version management system on daily basis.
- All artifacts are under version management.
- Build and deployment to development environment are automated.
- Build status and broken builds are visible to all.
- Anyone in the team can start build at any point in time
- Testing partially automated - greater than 50% code coverage.

4.

- Automated build and test cycle every time a change is committed.
- Build metrics gathered, visible, and acted on.
- Builds are not left broken and code is not committed on a broken build.
- Deployment to Test environment is automated.
- Non-functional testing is automated.
- Testing is automated as much as is practical.
- Environments can be provisioned at on demand.

5.

- Build and deployment are automated as much as is practical.
- The build pipeline extends directly into production.
- Tests are run in parallel across multiple machines.
- Trunk-based development is the standard practice and integration happens continuously. Branches are rarely created and are short lived.
- Check-ins occur multiple times each day.
- Environment & Infrastructure specifications are managed and versioned along with all other artefacts.
- The CI build creates and provisions environments to allow scalability for testing.
- Release and Change Management fully integrated into deployment process.
- The product is always in releasable state.
- Automated release process, release to customers is a business decision.

INCIDENT MANAGEMENT

1.

- The process for handling problems and incidents is ad-hoc.

2.

- An incident management process is in place and understood.
- Key people are identified for incident management

3.

- Incidents and problems feed into backlog
- Operational requirements are identified and tracked in the backlog.
- Problem management includes an urgency to address root cause
- Root cause analysis is consistently performed.
- The number of handoffs and teams involved in incident resolution is minimal.
- Level 1, 2, 3 support structures in place
- Instrumentation (monitoring) in place

4.

- Defects are resolved once and deployed automatically across all environments
- Defects resolution includes full regression tests
- System health is proactively monitored
- Root cause analysis is highly valued and regularly trended.

5.

- Feature Teams do own incident management
- Fail forward (failing in a way that enables you to identify and overcome underlying problem, encapsulate the way forward and reduce the likelihood of failure next time around)
- Team actively manages, monitors and reviews what happens in production, feedback loop is enabled and acted on.

RISK AND ISSUE MANAGEMENT

1.

- Risk management does not exist or is just a box to tick in order to get through a process (i.e. something done to keep risk and audit people happy).
- There is no awareness or transparency of current risks and issues in the team.

2.

- Risks have been identified and are captured using an appropriate artefact such as a risk story wall or risk register.
- Each identified risk has been assigned a risk mitigation or action plan.
- Risks are discussed as part of the iteration planning process.

3.

- Risks, issues and blockers are discussed and updated in appropriate detail as part of all sessions (stand-ups; iteration planning; showcases; steering committee meetings).
- Unmitigated or uncontrolled risks are easily identifiable
- Risks, controls and action plans are assigned to appropriate owners: someone who understands the risk, has responsibility and accountability for managing the risk, has interest in the risk, and has the authority to implement controls and actions.

4.

- The cost and benefits of risk mitigations and actions are evaluated and recorded before they are implemented
- Risk management roles, responsibilities and accountabilities have been defined and agreed upon.
- Regular check points with business stakeholders to ensure they are aware of all risks and issues

5.

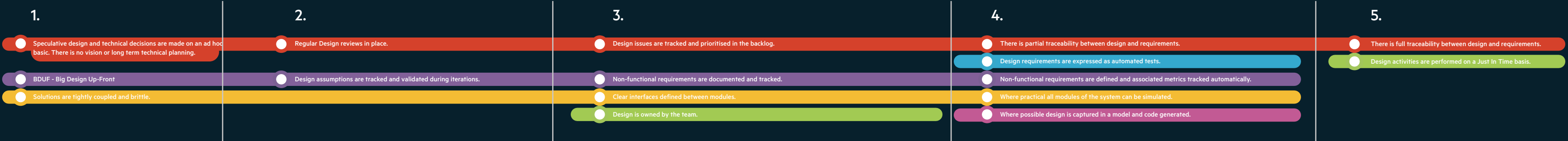
- Risk mitigations and actions are explicitly designed to increase positive outcomes and reduce negative outcomes
- The business participates in developing mitigation strategies
- Risk management is used to identify potential opportunities and drive strategy by thinking about risk in terms of uncertainty (which could be positive or negative)
- Measures and metrics exist to demonstrate the effectiveness of risk management and where improvement is required
- The financial impact of risk and issues is clearly understood



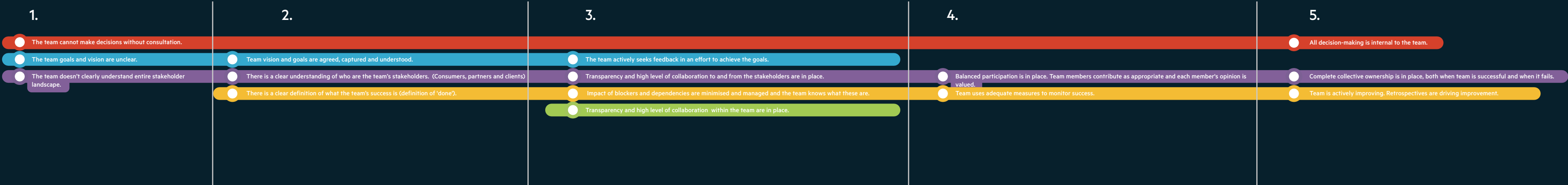
v1.0



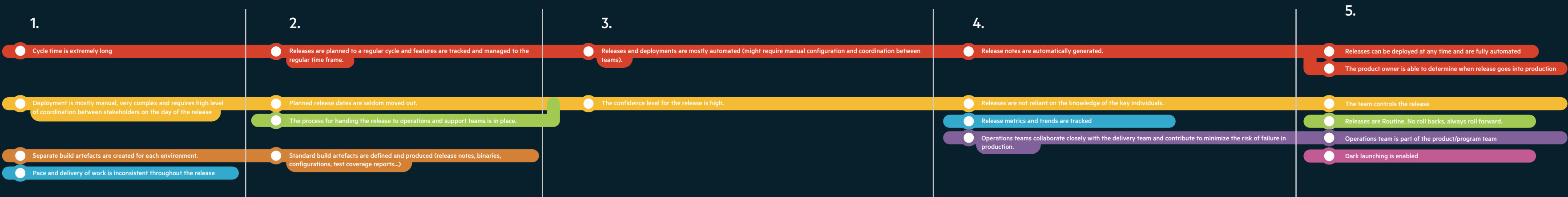
SOFTWARE DESIGN



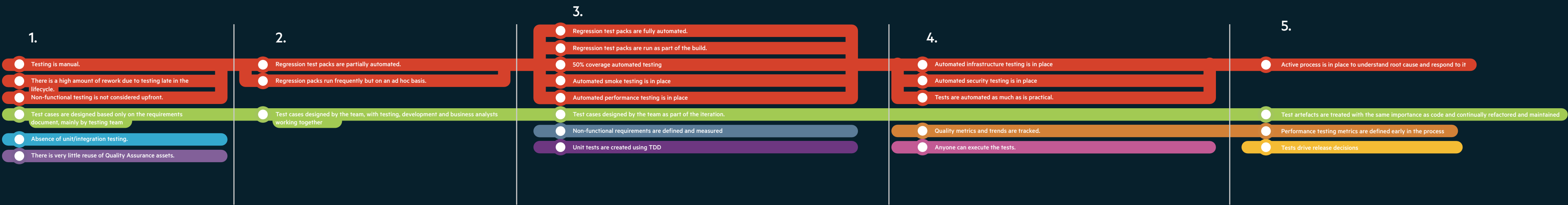
TEAMING



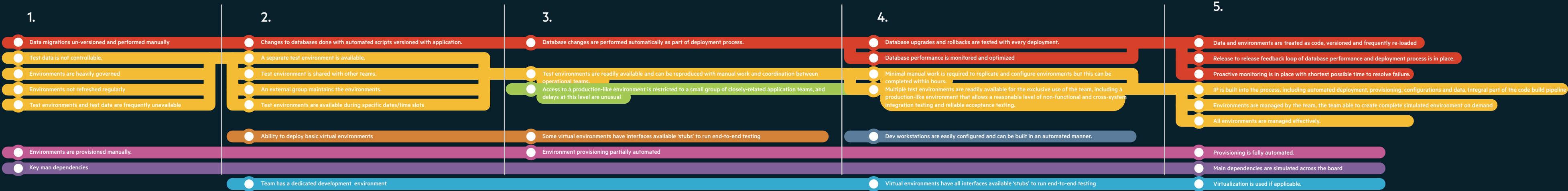
RELEASE MANAGEMENT



QUALITY ASSURANCE



ENVIRONMENTS



FEATURE TEAMS

