QA Maturity Model

The 5 levels:

1: Chaos

2: Awake

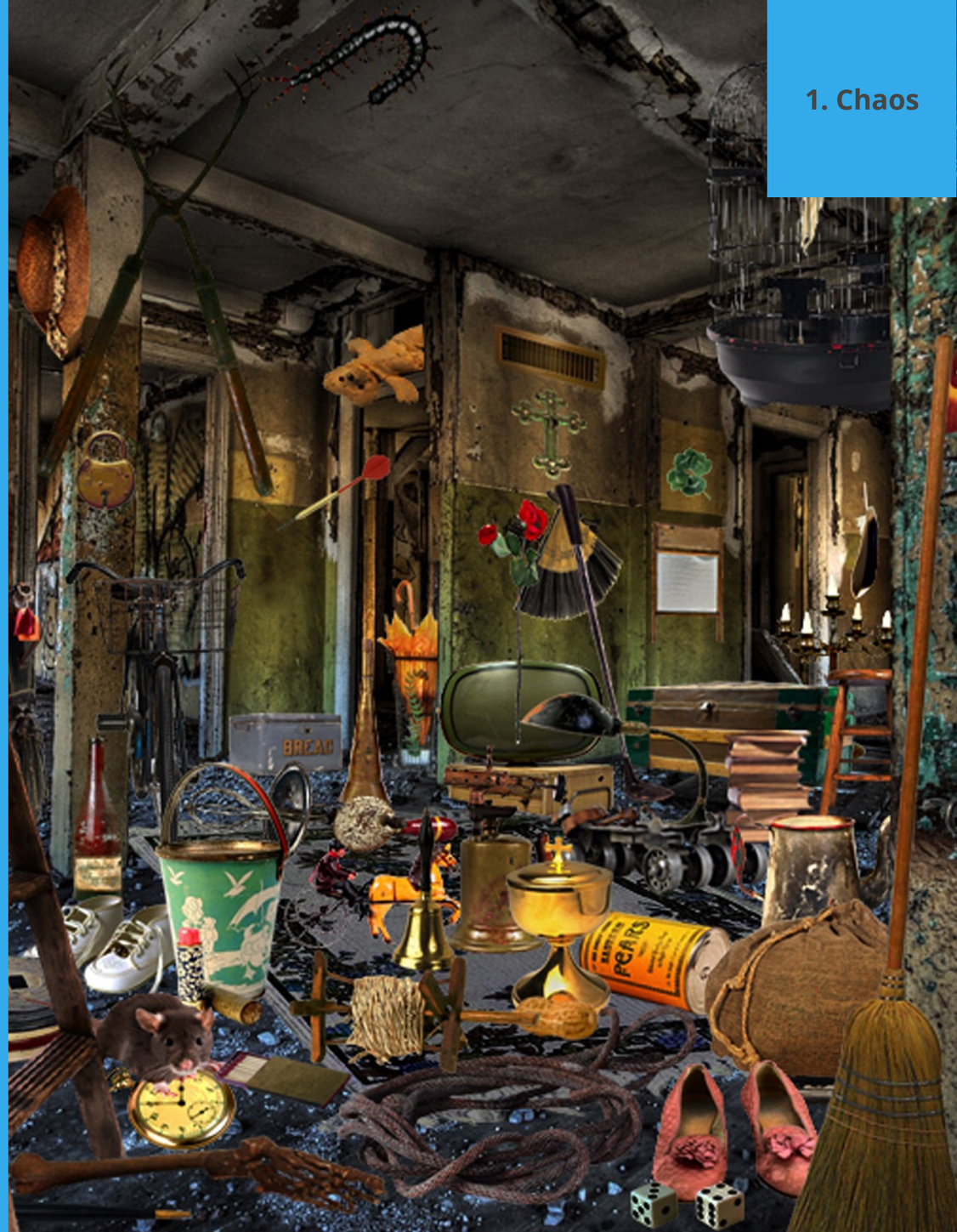
3: Grow

4: Collaboration

5: Transcendence

1. Chaos

Just few tests / No tests
Mostly manual tests
Many bugs in production
No automation knowledge
Skip failed tests
Skip tests to release faster
No pipelines / No CICD
Nobody is responsible for QA
No documentation
No code review / no code repository
A lot of time on maintenance
No time for developing new features
Micromanaging
No space to fail
Blame culture



2. Awake

Now we have a QA!
QA is the responsible for Quality
QA is a safety net to catch bugs
QA is driven to break the software
QA is not part of developers team
Environment with task managers
Status report environment / no autonomy
Regression tests are manual
Changes are not welcome
No testing strategy
CICD does not work properly
QA is a bottleneck to release
Hero culture

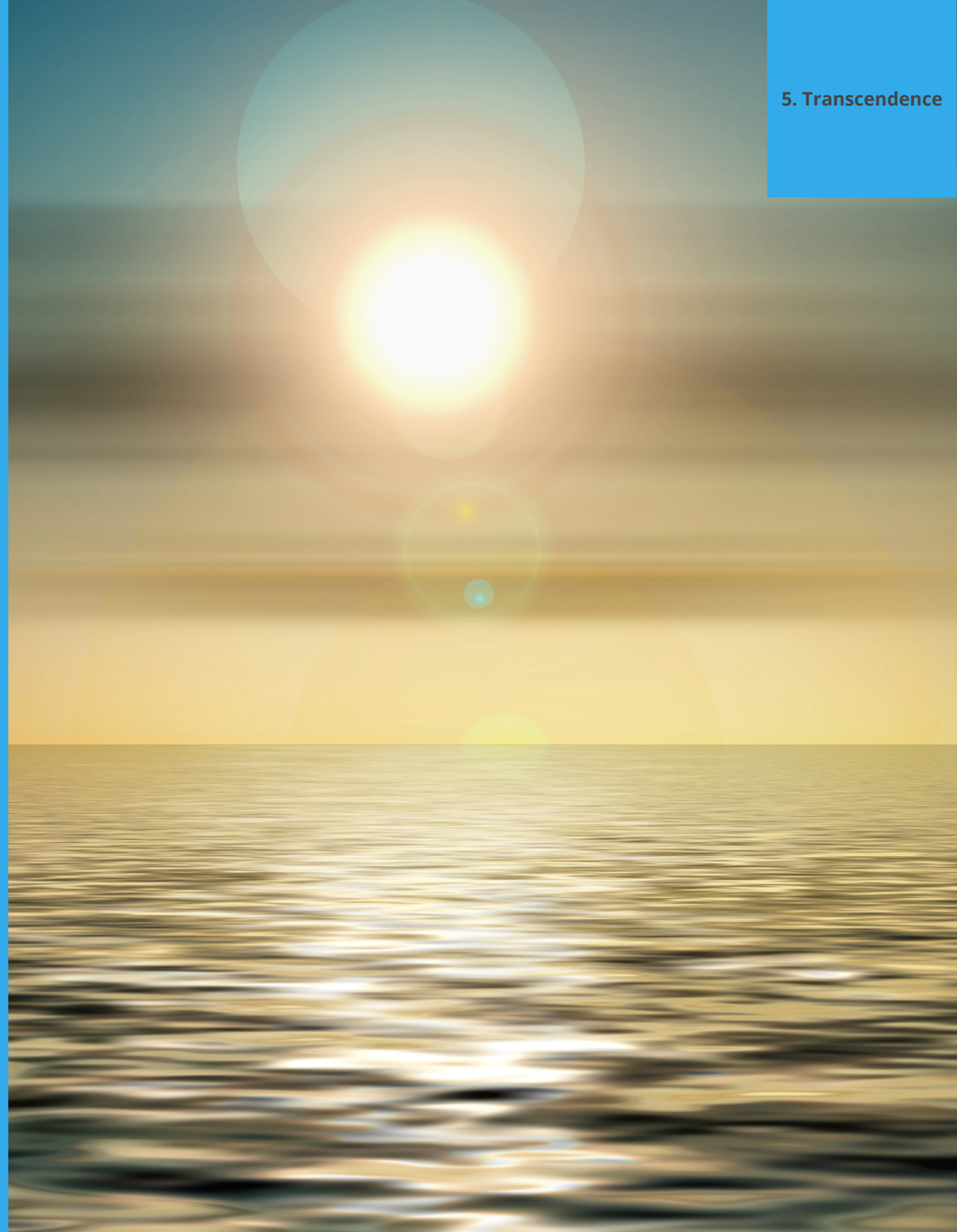
3. Grow

Creation of Community of Practices
ATDD / TDD / BDD being used
CICD is well defined
QA and Devs are part of the same team
QA and Devs are working closely
Pair programming / Pair Testing
Regression tests are automated
We automate tests for bug fixes
Flaky tests are fixed
Unit, integration, e2e, etc.
We learn from community
We share knowledge internally
DOJOs, tech talks, Hackathons, etc.

4.Collaboration

QA work as a quality coach/assistant
We have continuous delivery/deployment
Collaboration between Dev team and Business
Optimized Automated Regression Tests
We have all kind of tests that we need to have
QA helps on Continuous Improvement
QA helps to eliminate bottlenecks and waste
We have a healthy pace to deliver software
Changes are welcome
Leaders remove obstacles
Leaders create health environments
Leaders act like servant leaders

Everyone is responsible for Quality
We have a continuous improvement culture
We are driven to guarantee quality to our users
We work closely to business and customers
Changes are welcome to make customers happy
We have an environment with trust
We have an environment with high motivated people
We have technical excellence
We have self-organizing teams
We share knowledge to community
We have a fail fast culture
Agility and DevOps is spread



Key differences

	1 Chaos	2 Awake	3 Grow	4 Collaboration	5 Transcendence
Team	Nobody is responsible for QA No automation knowledge Mostly manual tests Many bugs in production	QA is the responsible for Quality QA is a safety net to catch bugs QA is driven to break the software QA is not part of developers team	Creation of Community of Practices ATDD / TDD / BDD being used QA and Devs are part of the same team QA and Devs are working closely	QA work as a quality coach/assistant Collaboration between Dev team and Business	Everyone is responsible for Quality Everyone is driven to guarantee quality to users We have an environment with high motivated people
Engineering	Just few tests / No tests No pipelines / No CICD Skip tests to release faster A lot of time on maintenance No time for new features No documentation	Status report environment No autonomy for engineer team Regression tests are manual No testing strategy CICD does not work properly QA is a bottleneck to release	Pair programming / Pair Testing CICD is well defined Regression tests are automated We automate tests for bug fixes Flaky tests are fixed Unit, integration, e2e, etc.	Continuous delivery/deployment Optimized Automated Regression Tests All kind of tests that we need QA helps on Continuous Improvement QA helps to eliminate bottlenecks and waste	Dev team works closely to business and customers We have technical excellence We have self-organizing teams Agility and DevOps is spread Automation is not a problem anymore
Culture	No space to fail Blame culture Micromanaging	Hero culture Changes are not welcome Task managers	We learn from community We share knowledge internally DOJOs, tech talks, Hackathons, etc.	We have a healthy pace to release Changes are welcome Leaders remove obstacles Leaders create health environments Leaders act like servant leaders	We have a continuous improvement culture, Kaizen culture, where we are always trying to improve what we have.