A practical framework: Four stages of software maturity

This framework helps CTOs to quickly diagnose software risk and organizational readiness, and align tech investments with reality.

Stage	What you'll see	What to do
Stagnation	Legacy systems, no inventory, hidden risk, unclear ownership.	Establish visibility across code, risk, and governance. Clarify roles and responsibilities.
Refactoring	Known issues and patchwork fixes persist; security issues and technical debt are increasingly visible but remain unprioritized.	Align fixes to business value, elevate buried risk, build roadmap buy-in across teams.
Modernization	Strategic roadmap forming, architecture under rework.	Strengthen architecture, scale metrics, connect modernization to outcomes with consistent processes.
Transformation	Mature architecture, trusted metrics, AI-ready processes and governance.	Operationalize monitoring, quantify impact, and align KPIs to business value.

Each stage calls for different priorities. The following overview describes what you'll see, and what to do next.

01 Stage 1: Stagnation

Most organizations we assess are here – operating with limited insight, driven by engineering priorities rather than business outcomes.

Typical signs

- IT is engineering-led, with budgets treated purely as cost-control tools.
- Focus on delivering features and fixing bugs, not addressing structural debt.
- Management-level insight into system quality, risk, and ownership is missing.
- Roles and responsibilities are unclear, with ownership of systems often fragmented across teams or individuals.

CTO priorities

- Establish visibility: baseline code quality, architectural health, and dependencies.
- Introduce measurable quality metrics across teams and systems.
- Focus on understanding before implementing major changes.

Stage 2: Refactoring

Organizations here have moved beyond firefighting, but structural improvements are still gradual and localized.

Typical signs

- Some structural improvements at the code level.
- Technical debt is being reduced, but fixes are localized.
- Limited architectural insight; changes are often reactive.
- Cross-team communication is ad hoc; success depends on individual champions rather than standardized processes.

CTO priorities

- Address root causes of recurring issues, not just symptoms.
- Build an improvement backlog tied to business outcomes.
- Enable cross-team conversations on architecture standards.

Stage 3: Modernization

At this stage, teams can tackle coordinated system-level change, though a lack of full business alignment still limits the overall impact.

Typical signs

- Coordinated modernization efforts at the system level.
- Ability to re-platform, re-architect, and replace outdated components.
- Process discipline improves; cross-functional groups ensure alignment at the system level.
- Executing major changes across multiple layers or systems simultaneously is still challenging.
- Business context is sometimes missing, so local optimizations don't always add up to the best enterprise-wide outcome.

CTO priorities

- Scale architectural standards across teams.
- Embed business context and process mapping into modernization work to ensure changes benefit cross-functional workflows and multi-system outcomes.
- Introduce governance that balances speed with quality.
- Connect modernization efforts directly to revenue, cost savings, and resilience.

Stage 4: Transformation

Few organizations are at this point: IT is a strategic partner to the business, driving innovation as much as enabling it. Getting here should be a north star for CTOs.

Typical signs

- IT and business have a shared understanding; strong governance ensures the organization knows where to invest, where impediments lie, what priorities to set, and which business growth opportunities to pursue through focused IT investments.
- Enterprise architecture actively drives innovation and market differentiation.
- Trusted, real-time insights available at the C-level, in decision-ready formats.

CTO priorities

- Continuously monitor quality, risk, and performance across the entire landscape.
- Invest in long-term resilience alongside speed.
- Use architecture as a lever for business model innovation.