bright ascension | HELIX®

SATELLITE SOFTWARE DEVELOPMENT PLAYBOOK

HOW TO GET TO ORBIT FASTER.

A practical guide to reducing development time



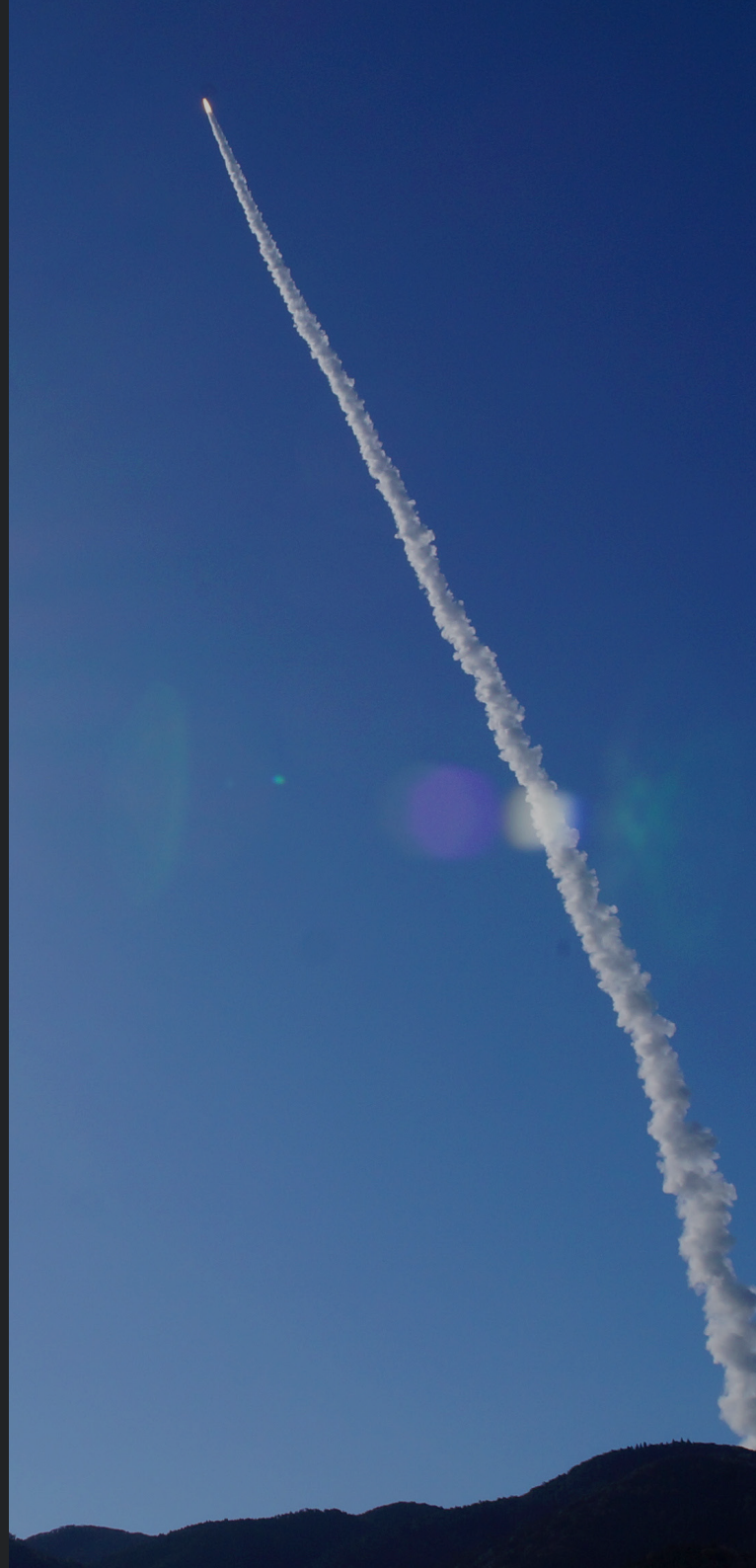
LEARN HOW REUSABLE, COMPONENT-BASED FLIGHT SOFTWARE CAN **REDUCE RISK**, **CUT DEVELOPMENT TIMELINES**, AND HELP YOU GET YOUR MISSION TO ORBIT **FASTER**.

HOW TO USE THIS PLAYBOOK.

This playbook is designed for spacecraft engineering & mission teams who need to accelerate development timelines without increasing technical or programmatic risk.

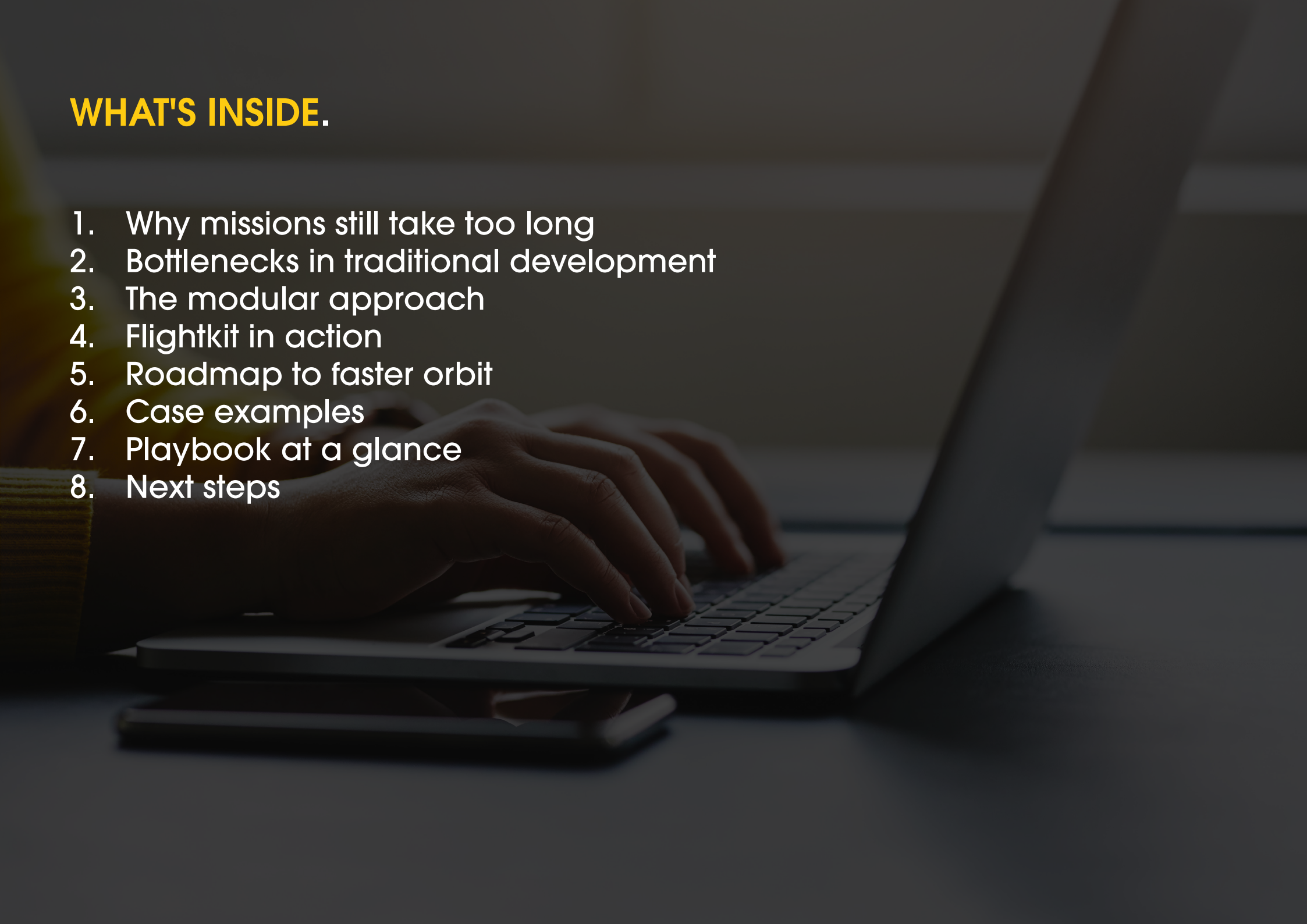
Use this playbook to:

- Identify where traditional flight software development slows your mission down.
- Understand how modular software architectures reduce complexity and rework.
- Apply a proven, step-by-step roadmap to your own mission planning.
- Benchmark what “good” looks like using real-world, quantifiable metrics.



WHAT'S INSIDE.

1. Why missions still take too long
2. Bottlenecks in traditional development
3. The modular approach
4. Flightkit in action
5. Roadmap to faster orbit
6. Case examples
7. Playbook at a glance
8. Next steps



1. WHY MISSIONS STILL TAKE TOO LONG.

THE COST OF TIME IN SATELLITE MISSIONS

Every satellite project is racing against the clock. Traditional mission development often spans multiple years, with teams investing significant engineering effort into bespoke flight software and one-off integrations.

Delays increase costs, push launch schedules, and introduce risk at every stage of the mission lifecycle. Late-stage integration issues, rework, and architectural changes are common — and expensive.

But there is a better way.

By adopting modular, reusable software architectures, mission teams can reduce development timelines, avoid unnecessary rework, and focus engineering effort where it delivers the most value: mission success.

2+ Years
typical development timeline.

<6 Months
with modular approach.

KEY TAKEAWAY

Small architectural changes can translate into months saved on your development timeline.

Start modular thinking early to reduce cost and schedule risk.

WHAT THIS MEANS FOR YOUR MISSION

Engineering teams spend less time re-implementing standard spacecraft functionality and more time delivering mission-specific capability.

Examples:
- Command handling.
- Telemetry.
- Fault management.

2. BOTTLENECKS IN TRADITIONAL DEVELOPMENT.

BESPOKE BUILDS, BIG BOTTLENECKS

Most satellite projects encounter the same recurring challenges:

- **Bespoke software development**
  Custom builds demand months of engineering time and extensive validation.
- **Integration delays**
  Late-stage conflicts often surface only once subsystems are combined.
- **Limited scalability**
  One-off software rarely adapts easily to future missions.
- **Hidden risk**
  Each rework introduces new uncertainty, cost, and schedule pressure.

These bottlenecks compound over time, often leading to missed launch windows or rushed late-stage decisions.

"Integration challenges are a common contributor to schedule slip."

KEY TAKEAWAY

Understanding where delays originate is the first step to eliminating them.

- Bespoke development frequently leads to significant rework during integration phases.
- Integration delays are a frequent contributor to schedule slip and missed launch opportunities.

WHAT THIS MEANS FOR YOUR MISSION

Without addressing structural software challenges, schedule pressure will persist regardless of team size or experience.

3. THE MODULAR APPROACH.

WHY MODULAR SOFTWARE WORKS

Modular software replaces bespoke development with pre-validated, reusable components that can be assembled to meet mission-specific requirements.

Rather than writing and validating the same functionality repeatedly, teams reuse proven building blocks — reducing integration complexity and development effort.

This approach enables:

- Independent development, testing, and validation of functionality.
- Reduced coupling between subsystems during integration.
- Reuse of proven behaviour across missions.
- Incremental system assembly rather than monolithic builds.

Feature	Bespoke	Modular (e.g. Flightkit)
Development time	Long	Short
Rework risk	High	Low
Scalability	Limited	Flexible
Integration	Manual	Streamlined

WHAT THIS MEANS FOR YOUR MISSION

Development effort is focused on configuring and extending proven functionality, rather than rebuilding core flight software from scratch.

UNDER THE HOOD: MODULAR ARCHITECTURE

Flightkit is Bright Ascension's modular flight software framework, designed to support rapid development and reuse across satellite missions. Modular architecture is designed to give engineering teams speed without sacrificing control or reliability.

At a high level, this architecture provides:

- Clear separation between core flight functionality and mission-specific logic.
- Well-defined interfaces between software components.
- Independent verification and validation of individual modules.
- Incremental integration without requiring full system rebuilds.

WHY THIS MATTERS

This allows teams to integrate, test, and evolve their flight software confidently as mission requirements change — without introducing system-wide risk.

4. FLIGHTKIT IN ACTION: MODULAR SOFTWARE PROVEN IN FLIGHT.

PROVEN MODULAR ARCHITECTURE FOR REAL MISSIONS

Flightkit applies modular software principles to real-world satellite missions, enabling faster delivery without sacrificing reliability.

KEY CAPABILITIES:

- Pre-built, mission-ready components that reduce engineering effort.
- Faster iteration cycles through independent testing and incremental integration.
- Scalable architecture that evolves with mission requirements.

PROOF POINTS:

- 60+ missions successfully deployed.
- Development timelines reduced to under 6 months.
- Up to 10× faster system build and configuration cycles.
- Up to 50% reduction in total mission cost.

KEY TAKEAWAY

Teams assemble only what they need, reducing effort, risk, and schedule pressure.

WHAT THIS MEANS FOR YOUR MISSION

You gain proven capability quickly, without locking yourself into rigid architectures.

5. ROADMAP TO FASTER ORBIT: A STEP-BY-STEP GUIDE.

A PRACTICAL APPROACH FOR ENGINEERING TEAMS

FOLLOW THIS ROADMAP: STREAMLINE DEVELOPMENT & REDUCE RISK

- **Define mission requirements and CONOPS early**
  Enable intelligent component selection and validate mission logic early.
- **Select pre-validated software components**
  Configure instead of rewriting core functionality.
- **Develop payload software independently**
  Use simulation or engineering models before final hardware availability.
- **Build integrations incrementally**
  Identify issues early and reduce integration risk.
- **Test continuously and localise risk**
  Validate components independently and minimise system-wide failures.
- **Reuse, iterate, and scale**
  Extend or replace components without redesigning the entire system.
- **Extend beyond launch readiness**
  Reduce commissioning time and improve operational readiness.

Faster time to orbit

=

Faster time to revenue

5. ROADMAP TO FASTER ORBIT: A STEP-BY-STEP GUIDE.

A PRACTICAL APPROACH FOR ENGINEERING TEAMS

CHECKLIST HIGHLIGHTS:

- ☑ Define requirements early - Avoid scope creep and late redesign.
- ☑ Select validated components - Reduce verification and testing effort.
- ☑ Develop independently - Decouple payload and platform development timelines.
- ☑ Incremental integration - Catch issues before system-level testing.
- ☑ Continuous testing - Localise failures and reduce debugging complexity.
- ☑ Reuse and scale - Build a foundation for future missions.
- ☑ Plan for operations early - Reduce commissioning delays post-launch.

KEY TAKEAWAY

Following a structured roadmap reduces uncertainty and accelerates readiness.

In commercial space, where revenue is generated only after a spacecraft is operational, reducing time to orbit directly translates into reduced time to market.

WHAT THIS MEANS FOR YOUR MISSION

Predictable progress replaces reactive problem-solving.

6. REAL TEAMS, REAL RESULTS.

QUANTIFIABLE OUTCOMES FROM MODULAR ADOPTION



Small satellite constellation
Reduced development timelines by up to 50%, accelerating launch readiness.

Earth observation mission
Eliminated integration rework, saving weeks of engineering effort.

First-time mission team
Built a mission-ready system in under 6 months with full pre-launch validation.

60+
Deployments
of validated flight software.

10x Faster
system build cycles.

Up to 50%
mission cost savings.

KEY TAKEAWAY

Modular software enables faster, more predictable mission delivery — regardless of team maturity.

7. THE PLAYBOOK AT A GLANCE.

THE MODULAR MISSION FORMULA

Replace bespoke builds with reusable components

Integrate incrementally

Test continuously

Reuse proven architectures

FLIGHTKIT BY THE NUMBERS

Up to 50% reduction in total mission costs

60+ spacecraft deployed

10× faster system build cycles

Development timelines of less than 6 months

8. NEXT STEPS: HOW TO GET STARTED.

YOUR MISSION DOESN'T NEED TO TAKE YEARS

FINAL THOUGHT

Modular software makes satellite missions faster, more predictable, and scalable for the future.

Take the next step toward faster, lower-risk mission delivery	Mini checklist
☑ Request a Flightkit demo.	☑ Are your requirements clearly documented?
☑ Download the technical specification.	☑ Have you identified reusable software components?
☑ Speak directly with our engineering team about your mission.	☑ Have you planned incremental integration and early testing?



GET TO ORBIT FASTER.

Building a spacecraft is hard — your software doesn't have to be. Flexible, reusable, well-supported software foundations help you launch faster, avoid delays and stay on budget.

READY TO TAKE THE NEXT STEP?

Request a free consultation to discuss your mission software needs.

brightascension.com