CLASSY LLAMA

Adobe Commerce SaaS vs PaaS: A Comprehensive Decision Kit for E-Commerce Leaders

Introduction

Choosing the right commerce platform deployment model is a critical decision for today's e-commerce leaders. In the Adobe Commerce ecosystem (formerly Magento), businesses now face a choice between a traditional Platform-as-a-Service (PaaS) model and a newer Software-as-a-Service (SaaS) model. This whitepaper is an educational decision kit designed for IT and marketing leaders – especially in retail, manufacturing, and automotive aftermarket industries – to navigate that choice. We focus on how each model aligns with goals of cost reduction and modernization, providing an objective look at the pros and cons of Adobe Commerce SaaS vs PaaS by scenario. You'll find a decision tree to guide your choice, a migration checklist, a TCO (Total Cost of Ownership) analysis worksheet, and a set of RFP questions to aid internal discussions. All insights are drawn from industry data and our experience as a long-time Adobe Silver Partner, Classy Llama, without a sales pitch – so your team can use this guide internally to make an informed decision.

Adobe Commerce SaaS vs PaaS: What's the Difference?

Adobe Commerce PaaS (Platform-as-a-Service) refers to the current Adobe Commerce (Magento) Cloud offering or on-premises deployment where you manage your own application code on a cloud platform. In this model, the merchant (and their IT/development team or partner) has full access to customize the core code, install extensions, and configure the environment within Adobe's cloud infrastructure. You maintain responsibility for code updates, version upgrades, and some infrastructure tuning (a shared-responsibility model for services like databases and search). Adobe provides the hosting platform and tools, but you control the application layer – making it a flexible option for highly customized stores. This flexibility is why PaaS has been attractive to companies with complex or unique requirements. However, it also means ongoing maintenance: custom code must be patched and upgraded regularly, and performance scaling may require active management.

Adobe Commerce SaaS (Software-as-a-Service) – known as Adobe Commerce "as a Cloud Service" (ACCS) – is the new fully managed, multi-tenant cloud version of Adobe Commerce. In this model, Adobe fully manages the core application and infrastructure, delivering a *versionless* platform that is always up-to-date. Merchants cannot change core code; instead they extend functionality through defined APIs, extension points, and Adobe's App Builder framework. The SaaS offering provides *continuous automatic updates* to features and security, so there are no more major upgrades to schedule. It promises *enhanced performance and scalability* out-of-the-box (with features like Edge Delivery for fast storefronts) and tighter integration with the Adobe Experience Cloud ecosystem. In short, SaaS trades some flexibility for simplified operations – Adobe handles the heavy lifting of hosting, updates, and scaling, allowing merchants to focus on business and content. It "moves Adobe Commerce from a PaaS model to a fully multi-tenant SaaS solution… solving long-standing pain points" of upgrades and infrastructure.

Summary: PaaS is like having a customizable platform hosted for you (great for control and unique needs, but you must maintain it), whereas SaaS is like a ready-to-use service (great for reducing IT burden and always getting the latest features, but within standard frameworks). Both models share the robust Adobe Commerce feature set – including support for B2C/B2B commerce, rich product/catalog management, and so on – but they differ in how you *implement, extend, and operate* the platform. The next sections will dive into the specific advantages and drawbacks of each approach, and in what scenarios each model makes sense.

Pros & Cons of SaaS vs PaaS by Scenario

Choosing between SaaS and PaaS often comes down to your business's specific scenario and requirements. Below, we outline the pros and cons of each model and highlight which types of businesses or project scenarios tend to benefit from each.

SaaS Model – Pros, Cons, and Ideal Scenarios

Key Pros of Adobe Commerce SaaS:

- **Reduced Maintenance & Automatic Updates:** Adobe's SaaS handles all platform updates, patches, and upgrades continuously in the background. This eliminates the need for costly upgrade projects and reduces the risk of falling behind on versions. For a merchant, that means a dramatically lower maintenance burden and no down-time for version releases. Over time, this *significantly lowers total cost of ownership* by removing one of the biggest historical cost drivers (upgrades).

- **Faster Time to Value & Modern Features:** Because it's an always-current cloud service, new features and enhancements roll out to SaaS users as soon as Adobe releases them. Adobe has promised a *feature-rich roadmap* on the SaaS platform – including advanced capabilities like AI-driven product recommendations and Generative AI content creation integrated into the platform[1]. For marketing teams, this means quicker access to modern tools (personalization, intelligent merchandising, etc.) without waiting for a lengthy upgrade. SaaS also natively supports modern architectures (headless commerce, PWAs) via GraphQL APIs and Edge Delivery Services for a faster, app-like storefront experience.

- **Scalability and Performance:** Adobe Commerce SaaS is fully cloud-native and *auto-scaling*. It offers real-time scalability and automatic security management[1]. Peak season traffic or sudden surges are handled by the platform without the merchant needing to provision extra servers. Adobe's Edge Network and CDN optimizations are built-in, promising global performance. For industries like automotive aftermarket with massive product catalogs and traffic spikes during promotions, this elasticity is a major benefit. Performance tuning and capacity planning become Adobe's responsibility, not yours.

- **Lower Infrastructure & IT Overhead:** With SaaS, you no longer need to manage infrastructure or devops tasks. Hosting, server configuration, database management, and even routine maintenance like applying security patches are all included. This can translate to significant cost savings in IT personnel or managed services expenses. It also frees your technical team to focus on strategic development (like building new customer experiences) rather than platform "plumbing." For organizations with a lean IT team or those aiming to reduce operational costs, SaaS is appealing. One reason many brands are re-evaluating on-premise platforms is the high and unpredictable TCO due to infrastructure and support costs[2] – SaaS directly addresses that by offering a predictable subscription that covers these services.

- **Stability and Security:** By locking down the core code and only allowing extensions through approved mechanisms, the SaaS model can deliver greater platform stability. You won't run into issues caused by a badly coded plugin or custom core modification because such changes aren't allowed in SaaS. Adobe manages the environment to maintain high availability and applies security updates continuously, providing automatic security and compliance updates as part of the service[1]. In scenarios where uptime and security are paramount (e.g. a manufacturer's B2B portal serving dealers nationwide), having the vendor ensure this at the platform level is a big plus. It's essentially Adobe's SLA that covers performance, uptime, and security, reducing risk for the merchant.

Cons and Considerations of SaaS:

- **Limited Deep Customization:** The trade-off for stability is that you cannot customize the core application code on Adobe Commerce SaaS. Customizations must be done via *extensions (Adobe App Builder apps, APIs, and new UI SDKs)*. While these tools are powerful, they may not (at least initially) cover every scenario that could be done with core modifications. Complex or unique business logic might be challenging to implement if it falls outside the SaaS's provided extension points. For example, companies in aftermarket auto parts often require highly custom fitment logic or complex B2B pricing rules. If the SaaS APIs don't yet support those needs, you might have to find creative solutions or wait for Adobe's roadmap. In general, SaaS platforms can struggle with very specialized requirements – they cater to standard use cases ("lowest common denominator") by design[3]. If your differentiation relies on a custom workflow deeply integrated into the commerce engine, verify that SaaS can accommodate it through its extension framework.

- **Dependency on Vendor Roadmap:** With SaaS, you are inherently tied to Adobe's pace of innovation and bug fixes. You cannot delay an update you don't like, nor implement a feature on your own that Adobe hasn't provided a hook for. If Adobe's SaaS roadmap doesn't prioritize a feature you need, you could be stuck waiting[4]. For instance, if you require a new payment method or a niche shipping workflow that isn't supported yet, you must rely on Adobe to deliver it (or create an external integration via APIs). In a PaaS model, by contrast, your developers could build almost anything directly into the platform. Companies should consider how critical *autonomy* and control over features are to their business. That said, Adobe's strong investment in the SaaS platform suggests rapid delivery of new capabilities – but the flexibility to do it "your way" is inherently less than with PaaS.

- **Integration & Data Constraints:** Integrating a multi-tenant SaaS with your on-premise or third-party systems can require different approaches. Adobe SaaS provides robust APIs and an *API Mesh* for integrations, but direct database access or custom integration code running on the commerce server is off the table. Most integrations (ERP, CRM, etc.) will need to go through cloud APIs or Adobe's integration services. This is generally good practice, but if your current system relies on, say, direct SQL access or heavy file exchange, you'll have to rework those integrations for the SaaS environment. Also, data residency or sovereignty needs should be discussed – your data will live in Adobe's cloud (hosted by them in their regions). Adobe will handle compliance (PCI, GDPR, etc.), but highly regulated industries might need to confirm the SaaS deployment meets specific regulatory certifications or allows necessary audits[5][6]. In manufacturing or automotive, this is usually less of an issue than in, say, healthcare or government, but it's a consideration for any IT leader: you're handing over certain controls and must trust the vendor's compliance.

- **Potential Cost Model Differences:** While SaaS reduces operational costs, the licensing cost structure might differ. Adobe has not publicly released detailed pricing for the SaaS model at the time of writing. It's likely a subscription based on gross merchandise value (GMV) or revenue, similar to the current Adobe Commerce license, but potentially with new metrics (since it includes hosting and many services). For budgeting, you'll want to model out a 3–5 year TCO. In some cases, SaaS could have higher direct subscription fees than a PaaS license + DIY hosting – but still be justified by the elimination of other expenses (infrastructure, upgrades, etc.). The key is predictability: SaaS tends to be a fixed annual cost (with scaling clauses), whereas PaaS might incur surprise costs (emergency devops, hardware upgrades for peak, etc.). We'll explore TCO in a later section.

Ideal Scenarios for SaaS:

Given these pros and cons, SaaS is often ideal for businesses that prioritize agility, cost efficiency, and standardization over highly bespoke functionality. Scenarios include:

- *Mid-sized Retailers or Manufacturers with Lean IT:* If your team is small and stretched, and you'd rather have Adobe handle the heavy lifting of the commerce platform, SaaS is attractive. It allows your team to focus on content, merchandising, and customer experience rather than software maintenance. For example, a regional auto parts manufacturer that wants to launch direct-to-consumer sales quickly, without building an internal devops capability, would benefit from SaaS's speed and low overhead.

- *Rapid Deployment and Growth:* If you need to launch or replatform quickly and start benefiting from new features (like AI-driven product recommendations) right away, SaaS can get you there faster. It's essentially plug-and-play once your data and customizations (via APIs) are connected. Companies undergoing digital modernization to catch up with competitors might use SaaS to leapfrog ahead with modern tech (PWA front-end, etc.) in months, not years.

- *Cost-Conscious Replatforming:* Organizations looking to cut costs in their e-commerce operations will find SaaS appealing. By removing the unpredictable costs of infrastructure issues and upgrade projects, SaaS offers a more stable long-term cost. One analysis found that traditional Adobe Commerce implementations carried on average 42% higher setup costs and 24% higher operating costs than a comparable SaaS platform[7]. SaaS can drastically reduce those via operational simplicity. If your CFO is scrutinizing e-commerce ROI, SaaS provides a clearer path to lower TCO (with the caveat that license fees must be weighed against that – we'll do so in TCO worksheet).

- *Standard B2C or Modular B2B Needs:* If your business model aligns well with out-of-the-box commerce capabilities (catalog, cart, checkout, etc.) and only requires *moderate* customization, SaaS will likely meet your needs. Adobe's SaaS still supports B2B features (customer company accounts, shared catalogs, etc., since it's the same core product) – and these are configurable. If you can stick to configuration and minimal custom logic, you'll reap the benefits of stability and quick improvements. Marketing teams will appreciate being able to use built-in personalization, A/B testing (via Adobe Target integration, presumably), and other SaaS-delivered enhancements without needing a developer for every change.

PaaS Model – Pros, Cons, and Ideal Scenarios

Key Pros of Adobe Commerce PaaS:

- **Unlimited Customization & Control:** The biggest strength of PaaS is freedom. You have full control over the application stack – you can install any Magento extension or build custom modules that alter core behaviors. There are effectively *no limits on integration or customization*; developers have complete control at the application level. This makes PaaS ideal for complex and unique business requirements. For example, a global manufacturer with multi-level distributor pricing, or an automotive parts seller needing a custom parts compatibility engine, can implement those bespoke features directly into the platform. You are not constrained by what the vendor's API supports – if you need to bend the system, you can (with enough developer skill). B2B commerce scenarios in particular often demand this flexibility: contract-based pricing, custom workflows for quotes/approvals, intricate product configuration – all these can be achieved on a PaaS implementation because you can extend or override core commerce logic as needed[8][9]. In short, PaaS is *adaptable to your business* rather than forcing your business to adapt to the software.

- **Greater Autonomy (No Vendor Lock on Features):** With a PaaS platform, you are in control of when to upgrade and which features to implement. You won't be auto-updated; you choose your timing for new versions (within Adobe's support window). If there's a new feature Adobe releases that you're not ready to use or that might break a customization, you can hold off until you prepare. Also, if you identify a feature gap, you have the autonomy to build a custom solution or integrate a third-party system of your choice. You're not tied to waiting for Adobe to deliver a feature on their roadmap[4]. For some organizations, especially those who view their commerce operations as a competitive differentiator, this control is crucial. It allows prioritizing features based on *your* roadmap, not the vendor's. Additionally, data access is more direct – your team can run complex SQL reports on the database or perform custom data science tasks because you can access the environment at a low level, something not possible in a locked-down SaaS environment.

- **Integration Flexibility:** In a PaaS deployment, integrating with other enterprise systems can be more straightforward in certain cases. Since you control the environment, you can use custom integration middleware or direct connections. For example, if you have a legacy ERP that only communicates via a specific protocol or if you want to use real-time database replication for inventory, those patterns can be implemented on your platform. PaaS doesn't limit API calls or events – you can utilize Adobe's APIs or directly extend them. If needed, you could even bypass some Magento business logic to sync data in a very specific way. This flexibility can be vital for manufacturing businesses that often have complex back-end systems and need their e-commerce tightly woven into supply chain or dealer management systems. Moreover, you can host the platform in a region or manner that suits data residency requirements (even on-premise or private cloud if not using Adobe's hosting). Compliance and security configurations can be tailored to your needs (e.g., specific encryption modules, sovereign cloud hosting) – PaaS can accommodate scenarios where a multi-tenant SaaS might not be allowed or optimal[6][10].

- **Performance Tuning & Specialized Extensions:** Some businesses require fine-grained control over performance tuning. With PaaS, because you can select the infrastructure specs (CPU, memory, etc. if self-hosted or via Adobe's flexible sizing) and configure low-level settings, you can optimize the stack for your application's characteristics. You might leverage custom caching strategies, alternate search technologies, or other performance tweaks that in SaaS would be standardize by Adobe. Additionally, there might be specialized extensions (third-party modules) critical to your industry that integrate deeply with Magento's core – these can be installed on PaaS but likely not on SaaS (unless those vendors rewrite them as SaaS-friendly apps). For instance, some aftermarket automotive retailers use third-party catalog management add-ons or search engines tailored for parts lookup; these might only be available for on-premise installation. PaaS ensures you can still use those tools.

- **Proven Solution for Complex Cases:** Adobe Commerce (Magento) has a long track record as a PaaS/on-premise solution powering very complex commerce sites. If you have an existing Adobe Commerce implementation with heavy customizations that work well for your business, staying on PaaS (at least in the short term) avoids the need to re-architect everything. It's the devil you know – and you can migrate to SaaS gradually when ready. Adobe will continue to support the PaaS model "for the foreseeable future", so you won't be left in the lurch if you stick with it now. Some organizations may choose PaaS as an interim step: upgrade to the latest Adobe Commerce version on PaaS, adopt some of Adobe's new SaaS-connected features (like Live Search or Product Recommendations which are offered as SaaS microservices) within that, and then move fully to SaaS in a phased approach later. PaaS thus offers a more controlled transition if you're not ready to go all-in on SaaS immediately.

Cons and Considerations of PaaS:

- **Higher Operational Burden & Cost:** The flip side of control is responsibility. With PaaS, your team (or your solution partner) is responsible for managing updates, maintenance, and infrastructure tuning in Adobe's cloud environment. This means you need a robust DevOps process to apply security patches, perform upgrades (which can be substantial projects), monitor performance, and handle scaling (adding servers or resources for peak times). All of this translates to higher ongoing costs – whether internal (hiring and retaining skilled developers/engineers) or external (paying an agency or Adobe for managed services). Historically, Adobe Commerce (on-prem/PaaS) has been associated with elevated TCO due to these factors[11]. For example, one study showed Adobe Commerce implementations incur significantly higher implementation and operating costs than SaaS alternatives[7]. This is partially due to needing cloud orchestration, DevOps, manual testing for upgrades, etc. that SaaS would handle[12][13]. If cost reduction is a top priority, PaaS is usually more expensive over the long run, especially when you factor in the opportunity cost of slower innovation (spending time on maintenance means slower time-to-market for new features)[14].
- **Complex Upgrades and Technical Debt:** Adobe Commerce's flexibility can become a double-edged sword over time. Every few years (or more often), a new version or major update comes out, and upgrading a heavily customized PaaS deployment is non-trivial. Each custom module might require code changes to be compatible with the new version; the database needs migration; extensive QA is required to ensure nothing breaks[13]. These upgrade projects can take months and significant budget – diverting resources from new feature development. If an organization has accumulated a lot of custom code ("technical debt"), they might find themselves stuck on older versions or spending hundreds of thousands on upgrades regularly[14]. Meanwhile, SaaS competitors are continuously delivering improvements with

no effort required by their users. This was cited by many brands as a reason to consider moving off the old model – they felt the innovation pace slowed by maintenance. In essence, PaaS can be *complex to keep current*. If your organization is not disciplined about refactoring and following best practices, the freedom of PaaS can lead to a messy codebase that is expensive to maintain.

- **Need for Skilled Development Team:** To leverage PaaS properly, you need access to strong development talent (either in-house or via a partner like Classy Llama). Magento/Adobe Commerce is a powerful platform, but building and customizing on it requires expertise in its architecture (PHP, now also leveraging some Adobe microservices, etc.). B2B and manufacturing companies often have IT teams focused on ERP, and might not have deep e-commerce development skills on staff. They can absolutely run PaaS by partnering with an agency or hiring a team, but that is an additional cost and management overhead. With SaaS, while you still might engage a partner for initial implementation or custom front-end work, the ongoing need for a deep bench of developers is lower. PaaS essentially means building your own custom solution on top of the platform, which is a software project that never truly "ends." If you do have a capable team and see your platform as something you want to heavily tailor, PaaS is empowering; if not, it can be overwhelming.
- **Risk of "Over-Customizing":** Many merchants have learned the hard way that just because you *can* customize everything on PaaS doesn't mean you *should*. Over-customization or poor-quality code can introduce bugs, performance issues, and instability. We've seen scenarios where a well-intentioned customization to meet a niche requirement ends up breaking the checkout process or making upgrades impossible without rewrite. SaaS by design prevents this by disallowing core modifications, whereas PaaS relies on discipline and good governance. If your company lacks strong technical leadership to set development standards, a PaaS project can result in a fragile site (and finger-pointing between IT and marketing when something goes wrong). Essentially, the PaaS model's flexibility can lead to increased risk if not managed properly. This is less of a concern if you have an experienced partner or team following Magento best practices – but it's a cautionary point.
- **Longer Time to Deploy (Potentially):** While it's not universally true (a simple PaaS site can be launched quickly too), generally a PaaS project can take longer to implement than a SaaS project. Since SaaS provides more

out-of-box and managed components, it can expedite the launch if your requirements align. PaaS projects often involve setting up your CI/CD pipelines, configuring cloud environments, and possibly more extensive development to achieve requirements that aren't native. If time-to-market is critical (say you need to launch a new direct e-commerce channel in a few months for a seasonal opportunity), consider whether the added complexity of PaaS development will impact that timeline. There are cases where PaaS might actually be faster – for instance, if a needed feature is not available in SaaS, building it yourself might be quicker than waiting for the vendor. One source notes that if SaaS doesn't support a precise requirement, a PaaS approach could lead to a quicker launch in that scenario. But as a general rule, the *default* SaaS implementation (with minimal customization) is faster to launch than a heavily customized PaaS build.

Ideal Scenarios for PaaS:

PaaS remains a strong choice for businesses that require maximum flexibility and have the resources to manage it. Scenarios include:

- *Complex B2B and Unique Workflows:* If you're a B2B enterprise with complex pricing, quoting, or product structures, PaaS might be necessary to deliver all required functionality. Manufacturing and automotive aftermarket companies often fall here – e.g., support for dealer-specific pricing, or complex part compatibility logic that ties into vehicle databases. These are instances where the ability to craft custom modules is invaluable. PaaS has been the go-to for such complexity. As noted in industry analysis, SaaS platforms "rarely allow the flexibility required" for many B2B use cases[8], so a PaaS approach is often better suited until SaaS capabilities catch up.

- *Heavily Regulated or IT-Controlled Environments:* If your organization or industry mandates stringent control over data, infrastructure, or release cycles, PaaS may be non-negotiable. For example, if you require your platform to be hosted in a specific private cloud due to internal IT policy, or you need compliance certifications that a multi-tenant SaaS can't yet provide, PaaS gives you that control[6][10]. Some government suppliers or industrial companies prefer having their own isolated environment, even if it's Adobe-managed PaaS. Similarly, if you need deep integration on the infrastructure level (like connecting to legacy systems via VPN, custom hardware, etc.), PaaS can accommodate those scenarios more easily.

- *Established Adobe Commerce Users with Custom Investments:* If you are already on Adobe Commerce (Magento) with a lot of custom features that work for you, and you're not under extreme pressure to reduce IT costs, staying on PaaS might be the least disruptive in the short term. You can continue to leverage your investment while gradually refactoring for a future SaaS migration. Adobe's phased migration approach encourages current customers to slowly adopt SaaS components (like using SaaS-based catalog and search services, or moving custom code into App Builder microservices) over time. So, an ideal scenario is a large enterprise that says: "We will eventually go SaaS, but over the next 1-2 years we'll straddle both to avoid a rushed rebuild." PaaS allows that coexistence strategy.

- *Highly Differentiated Brand Experiences:* If your competitive edge is tied to delivering a very unique customer experience or proprietary functionality, PaaS might serve you better. For instance, a company that differentiates on a custom product configurator or bespoke subscription logic might find a vanilla SaaS too constraining. PaaS would let you build those unique features directly. In such cases, the value of differentiation outweighs the cost of extra maintenance. Just be sure the ROI is truly there – sometimes "unique" features sound good in theory but provide marginal benefit to customers. Use PaaS when those custom features are mission-critical to revenue or customer satisfaction.

In summary, there is no one-size-fits-all answer – it's about aligning the platform model with your business strategy, capabilities, and priorities. Next, we provide a decision tree to further simplify this alignment process.

Decision Tree: Choosing Between SaaS and PaaS

To help crystallize which model fits your organization, consider the following decision points. This decision tree poses key questions and suggests a direction based on your answers:

- 1. Do you need deep, custom functionality beyond standard e-commerce features? (e.g. complex B2B pricing rules, custom checkout logic, non-standard product configurations)

  → If yes, lean towards PaaS, which imposes *no limits on customization* and lets developers build bespoke features. The PaaS model is better suited for complex use cases that out-of-box SaaS features can't meet.
  → If no (your needs are largely met by default features or minor tweaks), SaaS

could be a great fit, simplifying your life by avoiding custom code. Many businesses find the packaged capabilities of SaaS (with minor configuration) sufficient for their online store.

- 2. How important is rapid innovation and time-to-market for new features?

  → If very important, consider SaaS. Adobe Commerce SaaS will deliver new features (like AI-driven merchandising, performance updates, etc.) to you continuously without any upgrade effort. Your team can leverage enhancements immediately, which can be a competitive advantage. In a fast-moving retail environment, this agility can outweigh the lack of custom tweaks.
  → If you prioritize full control over a slower pace (e.g., you prefer to schedule and test every change extensively before release, even if it means delaying features), PaaS allows you to manage your own update cycle. Some organizations in regulated sectors might need this control, or culturally you might be more comfortable with a measured, internally controlled innovation pace.

- 3. What is the state of your IT team and budget for platform maintenance?

  → If limited or cost-constrained, SaaS is likely the safer choice. It offloads maintenance to Adobe, meaning you won't need to invest as much in DevOps engineers, hosting, and continuous upgrades. This directly supports cost reduction initiatives – SaaS "drastically cuts TCO and simplifies ongoing operations," as Adobe notes[15]. It also reduces the risk of something going wrong due to lack of oversight, since the platform is managed by experts.
  → If you have a strong development/DevOps team and budget to support them, and you view the e-commerce platform as a core piece of software you want to control, then PaaS is viable. Organizations with large IT departments or a strategic commitment to owning the tech stack can leverage their resources to manage PaaS effectively. Just ensure you account for those costs in your TCO analysis – having the team doesn't make the cost go away, it just means you're willing to bear it for the sake of control.

- 4. Are you aiming to modernize your tech stack with microservices, headless architecture, and cloud-native tech?

  → If yes, SaaS aligns well. The new Adobe Commerce SaaS is inherently

cloud-native and supports a headless approach (it even provides an Edge Delivery front-end and can integrate with your CMS or Experience Manager). It encourages breaking customizations into microservices (via App Builder) and using APIs. Essentially, SaaS will push you into a *modern architecture* that is future-proof. If modernization is a key goal (perhaps to improve site performance, or to better enable omnichannel experiences), SaaS is a catalyst.
→ If you have a legacy integration or monolithic approach that can't be changed immediately, PaaS might be needed in the interim. PaaS can still support headless and modern tech (you can deploy Adobe Commerce PaaS in headless mode with Adobe's PWA or a third-party front-end), but it will also let you run any legacy processes until you're ready to cut them over. This is more of a transitional reasoning – not so much choosing PaaS *over* SaaS, but using PaaS to bridge to a modern architecture gradually.

- 5. What are your uptime, support, and compliance requirements?

  → If you need a *guaranteed high uptime SLA, 24/7 platform support, and out-of-the-box compliance* (PCI, GDPR, etc.), SaaS will cover these as part of the service. Adobe's multi-tenant cloud is designed for reliability and they bear the responsibility for meeting SLAs. This can be reassuring for businesses where downtime directly equals lost revenue (think: an automotive parts site where even an hour of downtime could disrupt many customers worldwide). Also, for compliance, Adobe SaaS will handle updates to maintain compliance (like adapting to new security standards).
  → If you require a very customized security posture or environment (for example, integration with an on-premise identity management system, or hosting in a specific country for regulatory reasons), PaaS gives the flexibility to configure that. Some companies might also have internal policies that prefer having direct access to servers for auditing. In such cases, the PaaS environment – which can be single-tenant and more configurable – might be necessary. Keep in mind, Adobe Commerce PaaS on their cloud still gives you many compliance certifications, but you have more levers to pull if needed (like additional encryption modules, custom logging for auditors, etc.).

- 6. Is long-term total cost of ownership (TCO) a top deciding factor?

  → If TCO reduction is a top priority (i.e. you need to justify the platform choice financially over 5+ years), lean toward SaaS. By removing infrastructure and upgrade costs, SaaS often comes out ahead in TCO calculations, despite

potentially higher annual license fees[7]. We'll illustrate this in the next section's worksheet, but unless your SaaS subscription quote is exceptionally high, the operational savings usually tip the balance. This is especially true if you consider opportunity cost – the revenue lost or delayed due to slower feature delivery on PaaS[14].
→ If TCO is secondary to achieving specific capabilities (i.e. you're willing to spend more for a solution that fits perfectly), PaaS could be justified. Some businesses will invest in a higher TCO if it means unlocking revenue streams that would be impossible otherwise. For example, if a custom B2B portal built on PaaS drives millions in sales that a SaaS out-of-box site couldn't capture, the extra cost is worth it. Just be sure to document those assumptions and check that a SaaS approach truly can't meet those needs through configuration or slight process changes.

This decision tree should give you a general sense of direction. In many cases, your decision may not be a stark either/or – some companies might plan for a hybrid approach (e.g., continue on PaaS while gradually adopting SaaS services, or run certain brands on SaaS and others on PaaS). Adobe's ecosystem allows some flexibility here, but the core platform choice is binary for a given implementation. Once you have a leaning from the decision framework, the next steps are planning the migration (if moving to a new model), calculating TCO, and preparing the right questions for vendors and partners – which we cover next.

Migration Checklist: From PaaS to SaaS (or Vice Versa)

If you decide to migrate from one model to another – for instance, moving from an existing Adobe Commerce PaaS implementation to the new SaaS offering – it's essential to have a thorough plan. Below is a migration checklist to guide your internal team. This assumes a migration to Adobe Commerce SaaS (the more common scenario expected in 2025 and beyond), but many steps also apply if moving from a non-Adobe platform to Adobe Commerce SaaS/PaaS.

1. **Project Planning & Stakeholder Alignment:** First, ensure you have executive and stakeholder buy-in for the migration. Define the objectives (e.g., cost reduction, better customer experience, etc.) and success metrics. Establish a realistic timeline and budget. SaaS migration might be faster than a replatform to a completely new product, but it still requires significant work and coordination. Determine if you will take a phased approach or a "big bang" launch. Adobe recommends a phased migration for complex stores – e.g., gradually moving features to SaaS using the *Commerce Optimizer* tool

and interim integrations. Align on whether you'll run the old and new systems in parallel for a time or cut over at once.

2. **Inventory Your Current State:** Conduct a comprehensive audit of your current e-commerce implementation. Document all customizations, extension modules, integrations, and data flows. What third-party extensions are installed in your PaaS Magento? Which of those are essential vs. no longer heavily used? Create a list of all custom code/custom modules with descriptions. Also document your current storefront approach (Luma theme, custom theme, PWA Studio, etc.) because this will affect migration of the front-end. Capture details of integrations: ERP sync jobs, payment gateways, shipping providers, analytics scripts – everything that connects to your commerce platform. This inventory will serve as the foundation for planning what needs to change or rebuild in SaaS.

3. **Data Migration Plan:** Plan how you will migrate data from the old platform to the new one. Adobe provides a Bulk Data Migration Tool for moving from PaaS to SaaS. Key data includes products (catalog), customers, orders, content (pages, CMS blocks), and possibly promotion rules, etc. Identify any data that might not cleanly map to the new environment. For example, if you have custom database tables used by custom modules, those won't exist in SaaS – you might need to migrate that data to a new microservice or external store if still needed. Start with aligning your product catalog structure: SaaS will use the same core data models, but if you plan to adopt the new SaaS *Catalog Service* (part of Commerce Optimizer), you may need to import data into that service and ensure it matches your expectations. Also decide how much historical data to bring over (all orders from the past? or just last few years for performance?). *Tip:* do a trial migration of a subset of data early in the project to uncover any surprises.

4. **Extension and Custom Code Replacement:** For each customization identified in step 2, determine the approach in the SaaS model. Essentially, categorize your customizations into: Rebuild via App Builder (in-process custom code must be turned into out-of-process services on App Builder), Use Out-of-Box (perhaps the SaaS platform now offers a feature that your customization provided, meaning you can drop the custom code), Replace with Integration (some things might be handled by connecting to an external system instead of custom coding in the commerce platform), or Retire (some features might not be needed going forward). Adobe's SaaS encourages moving custom logic to *Adobe App Builder and API Mesh* services, so plan for that development. If you have third-party extensions, check if the vendor offers a

SaaS-compatible version or similar functionality as a service. Many Magento extension makers are likely creating Adobe App Builder apps for the new SaaS. If an extension (e.g., an advanced CMS or search tool) is not available, you might integrate a third-party SaaS product for that capability. Prioritize high-impact customizations – how will you achieve them in SaaS? This step can be the most time-consuming, as it involves actual development work to re-implement features in a new way.

5. **Integration Strategy and API Usage:** As noted, integrations need to be rethought for SaaS. For each external system (ERP, PIM, CRM, payment provider, etc.), decide how it will interface with Adobe Commerce SaaS. Leverage Adobe's API-first architecture – the SaaS offers both GraphQL and REST APIs for virtually all operations, and Adobe's API Mesh can help combine multiple APIs for easier consumption. Identify if any existing integration used direct database access or file exchange; those should be replaced with API-driven processes or Adobe's *Integration middleware (Integration Starter Kit)*. Also, set up the necessary authentication/credentials for API access to SaaS (OAuth tokens or Adobe IMS). If your integration is complex, consider using Adobe I/O Runtime (App Builder) to host integration scripts that respond to events from Commerce (for example, an order placement event triggers an App Builder function to send the order to your ERP). Adobe provides an *App Builder Eventing Framework* and *starter kits* for common scenarios like ERP integration. Use those as accelerators.

6. **Front-End and Experience Migration:** Adobe's SaaS includes a new approach to front-end: Commerce Storefront on Edge (aka Adobe's PWA on the Edge network). If your current site uses the old Luma theme or a traditional Magento theming approach, you will need to migrate to the new storefront technology (since SaaS likely won't support server-side PHP templates as in the past). Plan a front-end rebuild or refactoring: either adopting Adobe's recommended PWA/Edge Storefront or integrating your own headless storefront if you have one. If you already use Magento's PWA Studio or a headless front-end, that's a plus – you might be able to continue with it (Adobe SaaS can work with headless front-ends via APIs). However, Adobe is heavily promoting their integrated Edge storefront for performance benefits, and using it may simplify your SaaS implementation. This step involves front-end developers to recreate the design and UX on the new system. It's also an opportunity to do a design refresh if desired. Don't forget things like SEO considerations – ensure URL structures either remain the same or proper redirects are in place, to preserve your search rankings during migration.

7. **Performance & Load Testing:** Once you have the new SaaS environment set up with your data, custom extensions (App Builder services), and front-end, conduct thorough testing. Adobe's SaaS should handle scaling automatically, but you should still do load testing to simulate peak traffic with your specific customizations in place. This will help verify that your App Builder apps and integrations can handle volume and that any bottleneck is addressed before go-live. Test key journeys (browse, search, add-to-cart, checkout, account management, etc.) under load. Also test failure scenarios for integrations (e.g., if ERP is down, does the site queue orders properly?). Because you can't tweak the core infrastructure in SaaS, your performance tuning will largely involve optimizing your front-end, your App Builder code, and making sure you're using the platform features (like caching) correctly. Engage Adobe support if you find any issues; since this is a new model, Adobe might have performance best practices to share.

8. **Security & Compliance Check:** Ensure that the new environment meets any specific security requirements. Adobe SaaS will cover platform security, but you should review how things like admin access, SSO integration, and data policies are handled. For instance, if you integrate with an external user directory or have specific password policies, confirm those can be applied. Update any documentation like PCI compliance scope (your scope might actually reduce under SaaS, since you're not hosting the payment components yourself). If you have agreements with partners or customers about data handling, ensure nothing changes with the new cloud (e.g., data residency – know where Adobe hosts your data). Conduct a vulnerability scan or penetration test on a staging site if possible, to catch any application-level security issues introduced by your customizations.

9. **User Acceptance Testing & Training:** Run extensive UAT with your business users (both storefront and back-end admin users). Make sure that all critical business processes work as expected on the new SaaS site – from placing various types of orders (simple product, configurable, subscription if applicable, etc.) to processing in the admin (invoicing, shipping, etc.), and any integrations (ensure orders show up in ERP, etc.). This is the time for your marketing team to test their workflows too – e.g., can they update content easily, how do the new Experience Cloud integrations (like Adobe Analytics, Target, etc.) function, and are reports as expected? Document any changes in operational procedures (for example, maybe the way you schedule deployments is different now that code is in App Builder, or the method to clear cache is different). Train your staff on the new tools: Adobe Commerce SaaS might come with a new admin UI or at least new sections (for the

SaaS-specific features). Train developers as well, if you have internal devs, on the new development model (working with Adobe's cloud pipelines, App Builder, etc.). Adobe and partners often provide enablement sessions for teams migrating to SaaS – take advantage of those.

10. **Cutover Planning:** Finally, prepare for the go-live. Choose a cutover date and method: will you have a freeze on the old system and then do a quick data delta migration (e.g., bring over any orders placed in the last day during the cutover window)? Plan the DNS switch or URL redirect steps if the domain is staying the same. Make sure you have a rollback plan in case something critical happens (for example, keep the old site on standby in case you need to revert, though with data changes that can be tricky – often better to hotfix forward on the new site). Also, line up support: ensure your team or your solution partner and Adobe are all hands on deck during launch in case any issue arises. Monitor the site closely in the first few days/weeks for any error patterns, integration backlogs, etc. Many migrations have minor post-launch tweaks – be ready to address those quickly.

If you are migrating from another platform to Adobe Commerce (either SaaS or PaaS), many of the above steps apply as well, just that instead of converting custom code, you'll be mapping functionality from the old system to Adobe's features or extensions. In that case, add a step to sunset the legacy platform (data archival, SEO redirects from old URLs, informing customers of any changes like password resets, etc.).

Migrating to a new model is definitely a project, but with careful planning it can be a transformative success. Adobe's new SaaS offering even allows some intermediate states (for example, Adobe Commerce Optimizer can layer SaaS storefront and AI features on top of an existing platform as a stepping stone). Leverage these options if you need to reduce risk. The end goal is to smoothly transition such that your customers and business users only notice positive changes (faster site, new features) and not any disruptions.

TCO Worksheet: Evaluating the Total Cost of Ownership

A crucial part of the decision-making process is comparing the Total Cost of Ownership (TCO) for SaaS vs PaaS in your specific context. TCO includes *all* the costs over the lifespan of the platform – not just licensing or subscription fees, but also implementation, infrastructure, support, and opportunity costs. Below, we present a TCO evaluation framework. You can use this as a worksheet to estimate and compare costs.

According to industry guidance, key components of e-commerce TCO include[16]:

- **License/Subscription Costs:** This is the fee paid to Adobe (or any platform vendor). For Adobe Commerce PaaS (enterprise), this is typically an annual license fee based on GMV or revenue, often reaching six figures for large businesses[17]. For Adobe Commerce SaaS, expect a similar model (likely GMV-based subscription) which *includes* hosting and managed services. Compare the quoted fees for PaaS vs SaaS in your case. If you are on Magento Open Source (no license cost), then SaaS would introduce a license fee where there was none – but weigh that against savings elsewhere.

- **Hosting & Infrastructure:** For PaaS, if using Adobe's Cloud, some infrastructure is bundled, but often large merchants incur overages or need larger plans (thus higher fees). If self-hosting or using a third-party, this is the cost of servers, cloud services (AWS/Azure bills), CDNs, etc. Don't forget costs for dev/test environments too. SaaS includes all hosting/infrastructure in the subscription. In a TCO model, you'd likely mark the SaaS column as $0 (included) for this category, whereas PaaS might have a substantial annual dollar amount. Also account for scaling costs – PaaS might require over-provisioning for peak (spending more year-round to handle holiday traffic), while SaaS can scale on demand within its service.

- **Initial Implementation (Design & Development):** This covers the project to either implement SaaS or PaaS initially or during migration. This cost can vary widely based on how much customization you do. PaaS projects, especially replatforming or heavy customization, often have higher implementation costs[7] because of custom feature development, longer testing, etc. SaaS implementations might be lower if you stick closer to out-of-box capabilities. However, note that if migrating, even SaaS will have significant implementation effort (data migration, customizations via App Builder, etc.), so allocate budget for that. You may want to estimate the internal and external (agency) hours required and multiply by cost rates. For a fair TCO, amortize this initial cost over, say, 5 years (or whatever your typical platform lifecycle is).

- **Ongoing Development & Customization:** After launch, how much will you spend on continuously enhancing the site? In a PaaS model, ongoing dev might include building new features, installing and tweaking extensions, and adapting to changing needs. In a SaaS model, you might spend less here if the platform is delivering features for you (e.g., no need to custom-build an AI recommendation engine if Adobe provides one). However, you might reallocate development to more front-end and integration work. If you plan significant growth or changes, budget an annual amount for enhancements. Historically, Adobe Commerce (on-prem) clients have significant ongoing dev costs due to the complexity of projects[18]. SaaS could potentially lower that if fewer things need bespoke development. This is highly specific to your business – list anticipated initiatives and see if they'd incur dev work in either model.

- **Upgrades & Maintenance:** For PaaS, this is a *big* line item. How often do you upgrade (major version upgrades or minor version patching)? Each event can cost a lot in developer time, QA, possibly redoing custom code that's incompatible. Estimate an upgrade cost per year (or if you upgrade every 2 years, then half that cost per year for average). Also include routine maintenance: applying security patches, monitoring, performance tuning, etc. Often companies have a support retainer with an agency for this, or dedicated staff – include those costs. For SaaS, these costs are essentially zero on the platform side – no upgrade projects, and maintenance of the core is Adobe's job. You may still have some maintenance (e.g., updating your App Builder custom apps or front-end if Adobe changes an API), but those efforts should be far less frequent and less intense than a full platform upgrade. The Valtech insight underscores that eliminating upgrades *"significantly reduces the total cost of ownership"* for Adobe Commerce.

- **Support & Personnel:** Consider the personnel costs for supporting the platform. With PaaS, you might need a full-time (or multiple) developers, DevOps engineers, and QA working on the platform. Or you pay an agency for a support retainer. Also factor in things like training new developers, certification, etc. With SaaS, while it's not zero (you still might have a small team or at least an admin user who configures the site), the headcount can be lower. Perhaps you shift roles – instead of needing a system admin for servers, you might invest in a business analyst or merchandiser using the new AI tools. If you have an existing team, consider if SaaS allows you to reduce or reassign some roles (e.g., you might not need a dedicated Magento ops engineer in the long term). Any cost model should include salaries/contractor fees for the people involved in keeping the site running and improving it.

- **Opportunity Cost & Revenue Impact:** This is harder to quantify but important qualitatively. If a PaaS platform's limitations or lengthy development cycles cause you to miss market opportunities (e.g., a new feature took 6 months to develop, during which a competitor launched it in 1 month on a SaaS platform), that's an opportunity cost. SaaS proponents argue that faster feature delivery and performance improvements can lead to higher revenue (for instance, a faster site can improve conversion rates, and new personalization features can increase AOV). You might not put a dollar figure easily, but you can estimate potential uplift. For example, if the SaaS site's improved speed and features could boost conversion by X%, that additional revenue per year is essentially a negative cost (or a benefit) when comparing to status quo. On the flip side, if SaaS's lack of a certain feature would cause lost revenue (maybe you can't serve a niche segment as well), that should factor in favor of PaaS. In TCO, you can include a line for "revenue difference" if you have data – or simply account for it in a narrative way when making the case.

- **Intangibles:** Certain costs or savings are intangible but worth noting alongside TCO. For instance, risk mitigation: SaaS might reduce the risk of a catastrophic outage (since Adobe's SLA covers it), which could save you from rare but large losses. PaaS might give you peace of mind that you won't be forced into changes by Adobe (reducing risk of vendor-driven disruption). These don't go well in spreadsheets but are part of the business decision.

Once you list out these factors, you can create a comparison table. For example:

Cost Factor	Adobe Commerce SaaS (5-year)	Adobe Commerce PaaS (5-year)
License/Subscription Fees	$X (from Adobe quote, over 5 years)	$Y (Adobe license + cloud fees)
Infrastructure & Hosting	Included in subscription[1]	$Z (e.g., AWS hosting or Adobe overages)
Initial Implementation (one-time)	$A (estimate of migration project)	$B (if implementing/upgrading PaaS)
Ongoing Development & Enhancements	Lower ongoing dev (est. $C/yr, focusing on App Builder & front-end)	Higher ongoing dev (est. $D/yr, for custom features, etc.)

Cost Factor	Adobe Commerce SaaS (5-year)	Adobe Commerce PaaS (5-year)
Upgrade/Maintenance Projects	Essentially $0 for platform upgrades (none needed; maybe minor $ for app upkeep)	Significant (e.g., one major upgrade in 5 years $E, plus patch maintenance $F/yr)
Support/Staffing	Possibly reduced (e.g., 1 FTE admin + shared IT, $G/yr)	Higher (e.g., 2 FTE dev + 1 devops or equivalent agency retainer, $H/yr)
Total 5-Year TCO	calculate sum	calculate sum

(The above is a simplified illustration – fill in with your actual numbers.)

In many cases we've seen, Adobe Commerce SaaS is expected to reduce TCO by 20–30% or more compared to the equivalent PaaS setup, primarily by cutting the hidden costs of upgrades, infra management, and reducing opportunity cost through faster innovation. One external comparison (Adobe Commerce vs Shopify Plus) found Adobe's model to have 42% higher implementation and 24% higher operating costs historically[7] – precisely the inefficiencies Adobe is trying to eliminate with the new SaaS. While your mileage may vary, it's clear that if cost and efficiency are key, crunching these numbers will likely make a strong financial case for SaaS.

However, if your TCO analysis shows that PaaS costs more but also *earns more* (through capabilities that drive revenue), then it may still be the right choice. Always align the TCO discussion with expected business outcomes (e.g., "SaaS saves us $500k over 5 years and should also allow +5% revenue growth due to better UX, together improving our profitability significantly").

In summary, use the worksheet to quantify the differences. A platform decision is rarely purely financial – but having the financial picture clear helps ensure there are no surprises and that whichever path you choose is sustainable for the business.

RFP Question Set for Evaluating Platform Fit

When you're in the stage of evaluating Adobe Commerce SaaS vs PaaS – or comparing Adobe with other platforms – it's useful to have a set of questions to ask potential vendors or implementation partners. These RFP (Request for Proposal) questions ensure you cover the critical considerations discussed in this paper. Below is a curated list of questions that IT and marketing leaders should ask (either internally or to vendors like Adobe and their partners) to make an informed platform choice:

- Platform Model & Options:

- *Do you offer both SaaS and PaaS deployment options for the commerce platform, and what are the key differences in capabilities and responsibilities for each?* (For Adobe: how do Adobe Commerce SaaS and Adobe Commerce PaaS differ in terms of customization, hosting, and upgrade process?)

- *If we choose the SaaS model, what aspects of the system can we customize or extend, and via what mechanisms (APIs, app extensions, etc.)? Are there any limits or types of customizations that are not possible under SaaS?*

- *If we choose PaaS, what responsibilities will our team have versus the vendor?* For example, who manages uptime monitoring, scaling, applying security patches, etc. (Essentially understanding the shared responsibility model in PaaS vs the vendor responsibility in SaaS).

Scalability & Performance:

- *How does the platform handle scalability for peak loads?* (SaaS: Is auto-scaling handled transparently and is there any throttling or limits we should be aware of? PaaS: What tools or guidelines are provided for scaling infrastructure during high traffic?)

- *What performance optimizations are built-in?* For instance, ask about CDN usage, caching strategies, and for Adobe specifically, the role of Edge Delivery Services in accelerating content and storefront delivery.

- *Can the platform support our large catalog (e.g., tens of thousands or millions of SKUs) and complex product data efficiently?* If you're in aftermarket automotive with huge catalogs, ensure the vendor can provide references or benchmarks for similar size on SaaS vs PaaS.

Integration & Extensibility:

- *How well does the platform integrate with external systems (ERP, CRM, PIM, payment gateways, etc.)?* Is there native support or connectors for popular systems (SAP, Microsoft Dynamics, etc.), or will it rely on custom integration via APIs?

- *For SaaS: What integration tools or middleware do you provide?* (Adobe might mention API Mesh, App Builder, etc.). And ask *how data flows are handled securely* (e.g., are there webhooks, is there an event streaming capability for real-time sync?).

- *For PaaS: Do we get full database access and control for integration if needed?* And *what APIs does the platform offer out-of-the-box?* (Adobe has extensive REST and GraphQL – ensure any platform has a comprehensive API for all entities).

- *If we have existing custom integrations, what would the process be to implement them on your platform?* – This question can reveal if the vendor has done it before or if you'll be treading new ground.

Customization & Feature Development:

- *What capabilities exist for customizing the user experience and business logic?* (This is broad; for Adobe SaaS, expect answers about using their new UI SDK for front-end and App Builder for back-end custom logic).

- *Are there any known limitations for customizations in the SaaS platform?* For example, can we build custom checkout flows, or add custom product attributes and have custom pricing calculations? If certain things are not allowed in SaaS (like altering core checkout flow deeply), better to know upfront.

- *How do we implement advanced B2B features (like custom quoting workflows, tiered account roles, etc.)?* Are those features native to the platform, or do they require custom work? Adobe Commerce has a B2B module – ask if that is available and supported in SaaS as well.

- *Can we use third-party extensions or plugins?* If so, how are they managed or vetted in SaaS vs PaaS? (Adobe PaaS you can install any Magento extension; Adobe SaaS might have a curated marketplace or require rebuilding extensions as apps).

Security, Compliance & Data:

- *What security certifications and compliance standards does the platform adhere to?* (e.g., PCI DSS, SOC 2, ISO 27001, GDPR compliance). Adobe's cloud likely has these – get the specifics.

- *For SaaS: How is our data isolated in a multi-tenant environment?* (Often via logical separation, but you may ask for assurances on data privacy between tenants).

- *Where will our data be hosted?* (Important for international companies – ensure the data center regions align with your requirements or that the vendor offers EU data hosting if needed, etc.)

- *What is the disaster recovery and backup strategy?* For SaaS, Adobe should handle this (ask about RPO/RTO – Recovery Point and Time Objectives – how much data loss in worst case and how long to recover). For PaaS, see if Adobe (or your hosting) provides a DR solution or if you must handle it.

- *How frequently are security patches applied and how are critical vulnerabilities handled?* (SaaS will likely say immediately/continually; PaaS vendor might say they release patches and you apply them – ensure you know the cadence).

Operational Considerations:

- *What are the vendor's SLA (Service Level Agreements) for uptime and support response?* (e.g., Adobe might offer 99.99% uptime on SaaS with certain support response times for critical issues). Compare this with what you could achieve on PaaS with your resources.

- *What monitoring and analytics tools are provided for the platform's health?* (Do you get access to performance dashboards, error logs, etc., especially in SaaS where you can't log into the server?).

- *In the event of an issue or outage, what is the support process?* Is there 24/7 support and how do we engage (ticket, phone)? And if on PaaS, what support does Adobe give vs what your team/partner is expected to handle?

- *How are updates and new releases delivered?* For SaaS: confirm that updates are automatic and ask if there's any maintenance window or if it's seamless (Adobe might clarify if they do rolling updates, any downtime, etc.). For PaaS: ask about the frequency of releases and how long versions are supported (e.g., how often will we need to upgrade to stay supported?).

Cost Structure & Licensing:

- *Can you provide an estimated cost breakdown for our use case under SaaS vs PaaS?* (Adobe or a partner can often model this if you give them your volume metrics). Ensure the quote includes any cloud infrastructure fees for PaaS and compare to SaaS subscription.

- *Are there additional fees for certain features or usage?* For example, does using extra environments, or high API usage, or certain add-ons (like image optimization, search, etc.) cost extra? Adobe's new SaaS might bundle a lot, but clarify. Some SaaS platforms have caps on API calls or charges for additional storefronts, etc. Ask so you can avoid surprises.

- *What is the expected implementation effort and cost?* While not exactly a platform question, a vendor or partner's answer here helps gauge complexity. If Adobe SaaS is pitched as quicker, see if they have an estimate for timeline vs a typical PaaS project timeline.

- *How does the licensing handle multi-site or multi-brand scenarios?* If you run multiple storefronts (e.g., different brands or country sites), do you need multiple licenses or is it included? Magento Commerce traditionally allowed multiple websites on one license; check if SaaS retains that model.

Future Roadmap & Vendor Commitment:

- *What is the future roadmap for the platform?* For Adobe, this is crucial since they are transitioning from PaaS to SaaS. You'd ask: *Will new features be developed for both PaaS and SaaS, or primarily for SaaS going forward?* (We suspect mostly SaaS). *How long will PaaS be supported and do you foresee an end-of-life?* Adobe has stated they will support current PaaS "for the foreseeable future", but it's good to get any updated guidance.

- *How do you incorporate customer feedback or requests into the product roadmap?* This helps you gauge if the vendor will address any gaps you encounter. If you are considering SaaS but worried about a missing feature, know how that could be escalated or if there's a process for enhancements.

- *What is the ecosystem of partners and developers for this platform?* A strong ecosystem (agencies, extensions, community) can lower costs and risks. Adobe has a large community, but ensure that remains true in the SaaS era (e.g., are lots of partners skilled in the new SaaS tools?).

Migration Support:

- If you're migrating: *Do you provide tools or services to assist with migration from our current platform?* Adobe has migration tools – get details on what they cover (data migration, code compatibility scans, etc.).

- *Are there any incentives or programs for migrating to the SaaS platform?* (Sometimes vendors offer discount or funding for services if you migrate early, etc. Not guaranteed, but worth asking).

- *Can you connect us with references or clients similar to us who have made this transition?* Speaking to a peer company that moved from Magento PaaS to Adobe SaaS (or from another platform to Adobe SaaS) can provide valuable insight.

These questions will help you create a robust RFP or evaluation scorecard. By asking them, you force vendors to address the nuanced differences between SaaS and PaaS offerings, and you gather the information needed to make a well-rounded decision. The goal is to ensure no stone is left unturned – from technical fit to cost to future viability.

Remember that the answers should be documented and compared. For instance, if Adobe (or another vendor) answers these for both their SaaS and PaaS, you can directly compare side-by-side to see which aligns with your priorities. If you're comparing Adobe SaaS with a different SaaS (like Shopify Plus or BigCommerce) or a different PaaS, you'll see where each shines or falls short.

Conclusion

Adobe Commerce SaaS vs PaaS is not a trivial decision – it's a strategic choice that will impact your e-commerce operations for years to come. The SaaS model represents a modern, low-maintenance future with continuous innovation, ideal for companies looking to reduce IT overhead and accelerate growth with the latest features. The PaaS model, by contrast, offers maximum flexibility and control, suiting businesses with complex, unique needs and the willingness to invest in customization.

For IT leaders, the decision hinges on architecture control, integration needs, and long-term cost of ownership. For marketing and business leaders, it's about how quickly and reliably the platform can deliver the experiences and conversions you need, and whether it can adapt to your business strategies. Manufacturing and automotive enterprises, in particular, must weigh their intricate B2B requirements against the efficiency gains of modernization.

In many cases, we see a trend: cost reduction and modernization goals align strongly with the SaaS approach – Adobe's own advancements indicate that's the direction the industry is headed. The fully managed Adobe Commerce Cloud Service can drastically simplify operations and bring cutting-edge capabilities (AI, headless storefronts, etc.) to your fingertips. However, the PaaS route remains valid for those who need what we might call "bespoke horsepower" – the ability to fine-tune and extend every aspect of the commerce engine to drive a highly tailored business model.

Whichever path you choose, careful planning and expert guidance are key. Use the decision tree to introspect on your priorities, the migration checklist to map out the journey, the TCO worksheet to justify the investment, and the RFP questions to vet your options. As a final thought, keep in mind that this isn't necessarily a forever decision – some organizations may start on PaaS and transition to SaaS when the time is right (or vice versa, though less common). Adobe is building pathways to ease that transition, so you have options to remain agile.

Classy Llama, as a long-time Adobe partner, has navigated both models extensively. Our intent here wasn't to push one or the other, but to arm you with the knowledge to make the best decision for your company. With a clear understanding of Adobe Commerce SaaS vs PaaS, you can confidently choose the platform strategy that will reduce costs, modernize your commerce operations, and support your growth for the long run – all while avoiding pitfalls and surprises. Here's to your e-commerce success with the model that fits like a glove.

---

[1] Adobe Commerce SaaS - What we know so far

https://www.fluidcommerce.co.uk/blog/adobe-commerce-saas-what-we-know-so-far/21299/

[2] [7] [11] [12] [13] [14] [16] [17] [18] TCO on Adobe Commerce & Shopify Plus | Ecommerce Migrations

https://swankyagency.com/comparing-total-cost-of-ownership-adobe-commerce-shopify-plus/

[3] [4] [5] [6] [8] [9] [10] Why PaaS Outperforms SaaS in B2B eCommerce: an Expert Review

https://virtocommerce.com/blog/paas-vs-saas-ecommerce

[15] Accelerate growth and time to market with Adobe Commerce.

https://business.adobe.com/resources/sdk/drive-organic-growth-and-maximize-conversions-with-adobe-commerce.html