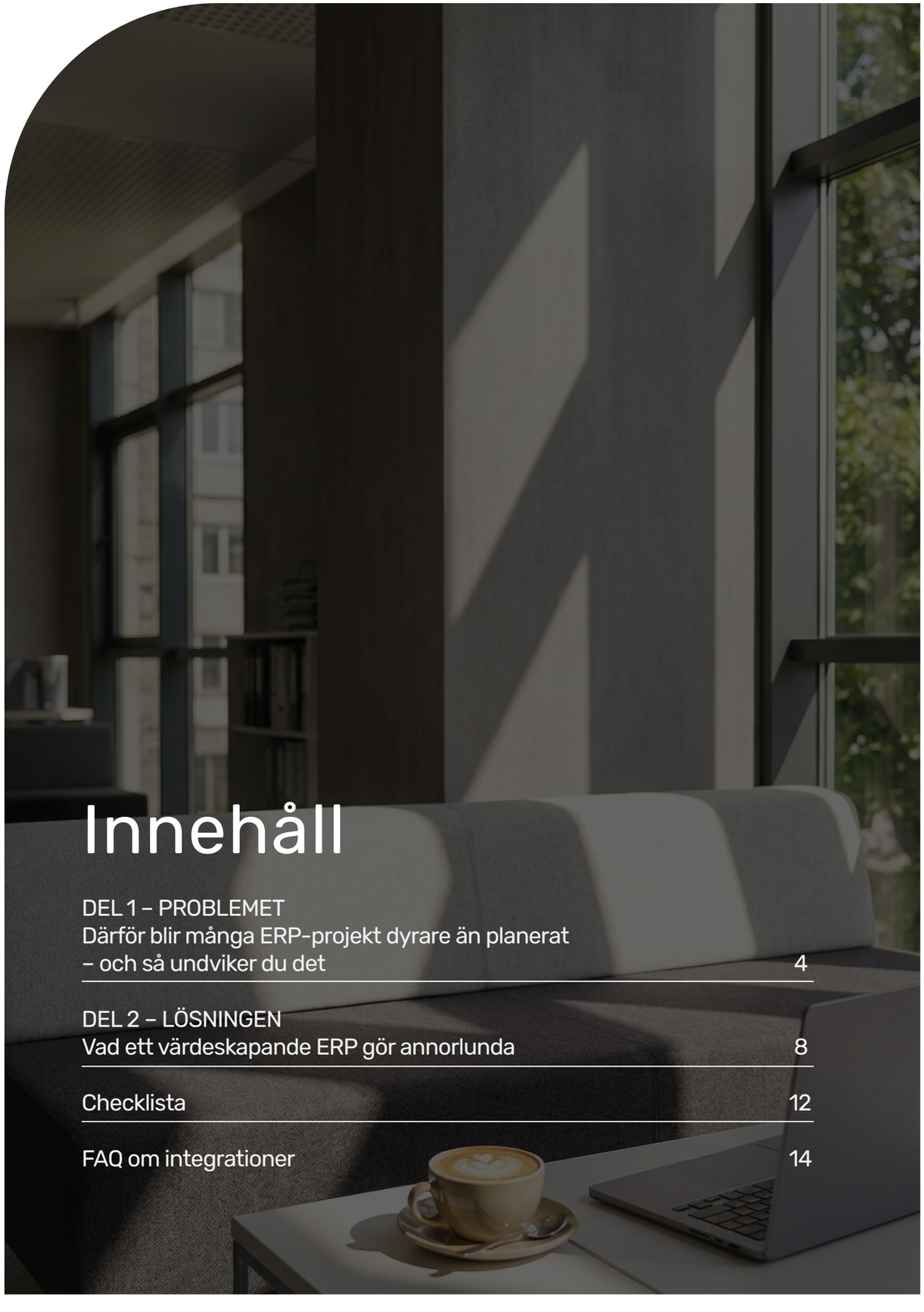
Business NXT
by VISMA

Integrationsguide för ledare

Så undviker du att ERP-systemet hämmar tillväxt, beslutsfattande och AI-initiativ



Innehåll

Ett modernt affärssystem ska inte bara stödja verksamheten. Det ska skapa värde i sig självt genom bättre beslut, högre tempo och nya affärsmöjligheter. Frågan är vad som faktiskt avgör om du lyckas med det.

Svaret är enklare än många tror: **Det handlar om samspelet mellan systemen.**

Det som avgör är inte funktionaliteten i sig, utan hur systemet hänger ihop med resten av verksamheten.

Ändå är det precis här många trampar snett: Integrationerna underskattas i beslutsfasen, och konsekvenserna märks först när systemet tas i bruk.

I den här guiden får du veta vad som faktiskt måste vara på plats för att affärssystemet ska bli en motor och inte en flaskhals. Vi visar hur den nya generationens ERP är byggda med integrationer **som en del av själva grundvalen**, och hur det förbättrar beslutsunderlaget och ökar er konkurrenskraft.

Har du bråttom kan du hoppa direkt till **checklistan** på sida 12. Men det lönar sig att lägga några minuter på helheten först.

DEL 1 – PROBLEMET

Därför blir många ERP-projekt dyrare än planerat – och så undviker du det

Det finns framför allt **tre strukturella orsaker** till att många ERP-projekt blir dyrare än planerat. Låt oss gå igenom dem snabbt innan vi presenterar lösningen →

1. Systemarkitekturen blir alltmer komplex och sårbar

Många ERP-system är byggda med flera hundra API-slutpunkter. Varje integration kräver sina egna kopplingar, sina egna anpassningar och sina egna kostnader. I praktiken innebär detta att ju fler integrationer du kopplar på, desto mer komplex och skör blir systemarkitekturen.

Kan detta undvikas? Ja, och nyckeln är att *samla all dataåtkomst i en enda API-slutpunkt*, så att du slipper hundratals separata kopplingar. Vi berättar mer om detta i del 2 på sida 8 om ett ögonblick.

2. Dold komplexitet upptäcks för sent

Det som på pappret ser ut som "en enda integration", visar sig ofta vara många. Du måste hämta data från olika ställen, sammanställa dem och validera i flera steg.

Resultatet blir förseningar, beroenden och oförutsägbarhet i driften.

Kan detta undvikas? Ja, och nyckeln är en arkitektur där data hämtas *samlat och färdigstrukturerat i en enda förfrågan*, istället för att sammanställas i efterhand. Mer om även detta i del 2 på sida 8!

3. Teknisk skuld samlas på hög från dag ett

För att få systemen att fungera tillsammans tar olika personer ofta ansvar för att få saker att rulla i vardagen. IT-avdelningen sätter upp tillfälliga integrationer, de ekonomiansvariga börjar exportera och justera siffror i Excel, affärsutvecklarna inför manuella rutiner för att täppa till luckorna mellan systemen … och så vidare.

Sådana tillfälliga lösningar blir snabbt en permanent del av vardagen. De växer och blir till en struktur som ingen har full överblick över, men som alla är beroende av.

Och det är vad vi kallar teknisk skuld: kostnader och extraarbete som uppstår när man väljer en snabb och enkel IT-lösning framför en mer solid och långsiktig lösning.

Kan detta undvikas? Ja, och nyckeln är ett system som kan anpassas efter dina behov direkt i datamodellen, utan tillfälliga "workarounds" och tredjepartslösningar.

Vi beskriver även detta i del 2.

Vad innebär dessa problem för företaget och dig som ledare?

Det kan inte sägas tydligt nog: Utmaningarna vi har beskrivit här är inte ett tekniskt problem. De utgör ett **affärsproblem**. När data inte flödar fritt mellan systemen påverkar det direkt hur verksamheten styrs:

- **Beslut fattas i efterhand** eftersom data måste hämtas och kvalitetssäkras manuellt.
- **Kostnaderna ökar eftersom varje integration är dyr att bygga, och ännu dyrare att förändra och förvalta.**
- **Tempot i organisationen sjunker** eftersom anställda jobbar runt systemen, inte med dem.

Det här är anledningarna till att många fortfarande ser affärssystemet som en ren kostnad. Det handlar inte om att systemet i sig är dåligt, utan om att det inte är byggt för att skapa flöde.

Men ett affärssystem kan vara så mycket mer än ett nödvändigt ont för att säkra kontroll och efterlevnad. Ett modernt ERP-system ger dig snabbhet, precision och bättre beslut. Skillnaden ligger i hur det är byggt.

DEL 2 – LÖSNINGEN

Vad ett värdeskapande ERP gör annorlunda

Vilka företag lyckas med integrationer från dag ett? Och vilka hamnar i ett träsk av onödig komplexitet, ökade kostnader och teknisk skuld? Skillnaden dem emellan ligger inte främst i vilka system de väljer, **utan i hur systemen är byggda för att prata med varandra.**

Ett värdeskapande ERP är utvecklat med integrationer som utgångspunkt, inte som ett tillägg. Det innebär några helt konkreta skillnader →

Ett API istället för hundratals kopplingar

I många traditionella affärssystem måste du förhålla dig till ett stort antal API-slutpunkter. Varje enskild process, varje enskilt dataset och varje enskild integration har sin egen ingång.

I praktiken betyder detta att du får **fler felkopplingar, fler felkällor och högre kostnader för varje ny integration.**

I modern arkitektur är däremot hela datamodellen tillgänglig via en enda API-slutpunkt. Det innebär:

- Du kopplar upp dig mot **en struktur**, inte många.
- Nya integrationer **bygger vidare på det som redan finns**.
- **Komplexiteten minskar** – även när systemlandskapet växer.

På så sätt är det möjligt att undvika den sårbara arkitektur vi beskrev tidigare.

Exempel från vardagen:
Avdelningen vill testa ett nytt rapporteringsverktyg och sätter IT-chefen på uppgiften, i hopp om att kunna börja använda det nästa månad. IT-chefen kan enkelt koppla verktyget till en och samma struktur, och avdelningen är igång inom ett par dagar.

Från många förfrågningar till en enda: **GraphQL**

I traditionella API:er måste data hämtas i flera steg: en förfrågan för en kund, en förfrågan för en order, en för lager, och så vidare – och sedan måste allt sammanställas i efterhand. Det är här mycket av den dolda komplexiteten uppstår.

Med GraphQL fungerar det fundamentalt annorlunda: Du hämtar all relevant data i en enda förfrågan, färdigstrukturerat.

Det ger tre konkreta fördelar:

- Du slipper hämta data från flera ställen och sammanställa dem manuellt i efterhand.
- Du minskar antalet anrop mot systemet dramatiskt.
- Du får betydligt snabbare responstider.

I praktiken betyder detta att responstiden går från märkbar väntetid till nästintill realtid. Den skillnaden utgör en enorm fördel. Den skillnaden utgör en enorm fördel.

Exempel från vardagen:
Ekonomichefen behöver snabbt få fram färska nyckeltal inför ett viktigt styrelsemöte. Istället för att hämta siffror från flera källor och dubbelkolla att allt stämmer, får hon allt samlat på två sekunder och kan gå in till styrelsemötet med en solid presentation.

Arbetet flyter smidigare och konkurrenskraften stärks

När dessa principer samverkar händer något fundamentalt: Hela organisationen slutar arbeta *runt* systemen och börjar i stället arbeta *med* dem. Eller för att uttrycka det i faktiska resultat:

- Data flödar **kontinuerligt** mellan systemen.
- Du kan fatta beslut baserat på **uppdaterad information**.
- Integrationer blir en **styrka**, inte en kostnadsdrivare.

Här har du också nyckeln till att öka intäkterna, minska utgifterna och ligga steget före konkurrenterna.

Byggt för integrationer från start: **API-first**

I många äldre system är API:et något som har lagts till i efterhand. I ett modernt affärssystem är API:et själva fundamentet. Vad innebär det?

Det innebär att **allt som finns i systemet också är tillgängligt via API:et.** Du får inte bara tillgång till delar av datan eller utvalda funktioner – du får tillgång till hela datamodellen.

Konsekvensen är enkel: Du slipper bygga egna lösningar för att komma till rätta med begränsningar i systemet. Det som tidigare krävde specialanpassningar, manuella processer eller tredjepartslösningar blir i stället en naturlig del av hur systemet fungerar.

Detta blir ännu viktigare inför den AI-era vi står på tröskeln till: Ju bättre och mer komplett API:et är, desto enklare är det att koppla på nya verktyg och agenter som kan använda datan **direkt – utan egna integrationsprojekt.**

Exempel från vardagen:
Ditt företag ska testa ett nytt AI-verktyg för prognoser. I stället för att behöva dra igång ett eget integrationsprojekt som tar flera veckor, kan verktyget bara kopplas direkt på datan och ge värde från första stund.

När systemet anpassar sig efter dig, inte tvärtom: **Flexibel datamodell**

En av de vanligaste orsakerna till tillfälliga “workarounds” som blir permanenta och bygger teknisk skuld är enkel: Systemet täcker inte företagets behov.

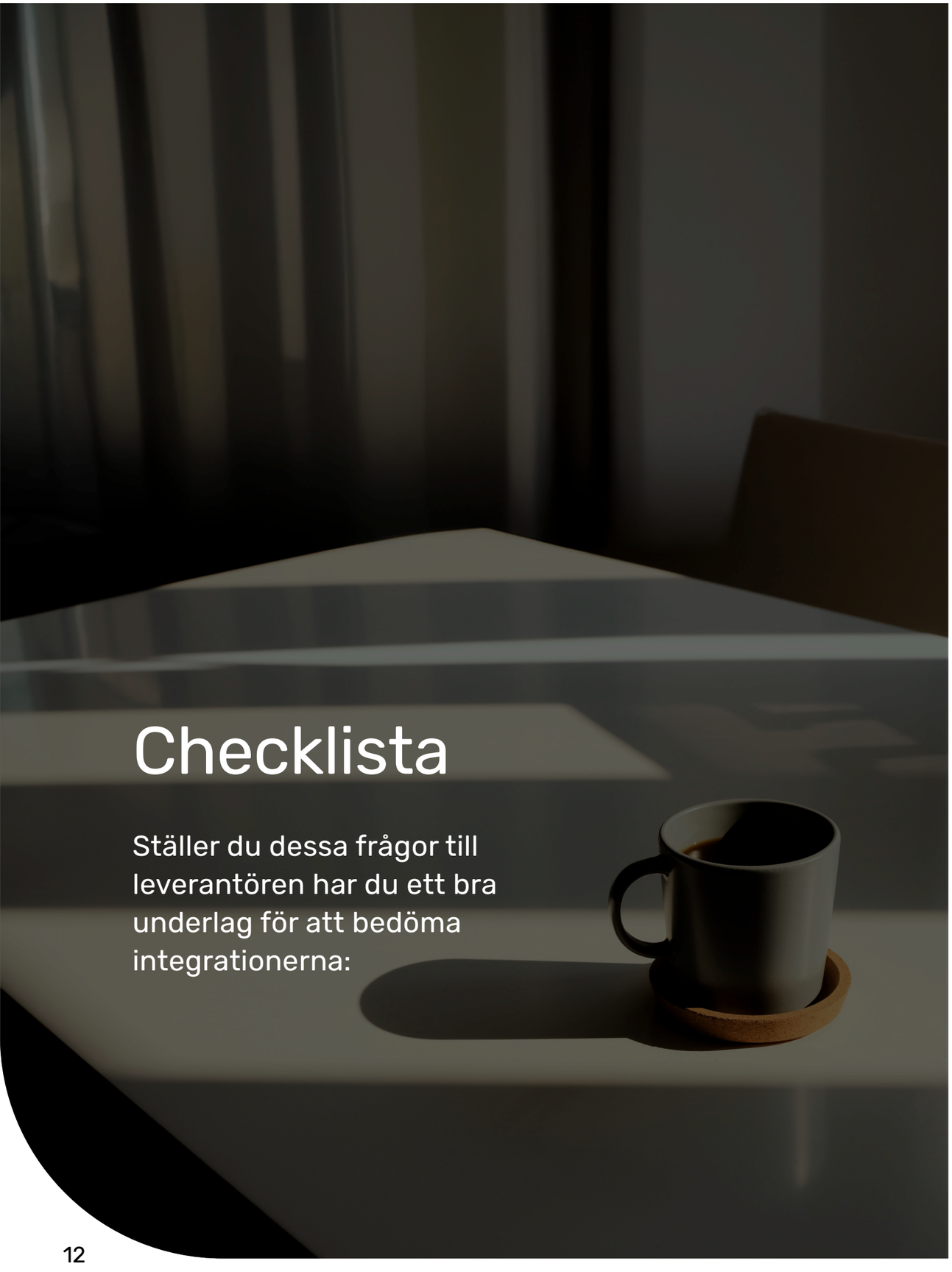
I traditionella lösningar innebär detta att du måste anpassa processerna, eller att du måste bygga något utanför systemet.

Med en flexibel datamodell får du en annan möjlighet: Du kan utöka systemet själv. Detta kallas *även data model extension* (DME). Med DME/flexibel datamodell kan du:

- Lägga till egna fält
- Skapa nya datastrukturer
- Anpassa lösningen för just ditt företag

Och viktigast av allt: **Allt du lägger till blir omedelbart tillgängligt i API:et.** Det betyder att du slipper till exempel Excel som mellanled, speciallösningar vid sidan av och manuella processer för att täppa till luckor. Detta är nyckeln till att undvika teknisk skuld från dag ett.

Exempel från vardagen:
Affärsutvecklaren på ditt företag behöver följa upp en helt ny intäktsmodell. I stället för att föra in detta i Excel vid sidan av, kan han enkelt sköta det direkt i systemet och få med det i rapporterna omedelbart. Han har alltså i själva verket utökat systemet, men utan att det var mer komplicerat för honom än att lägga till en kolumn i ett kalkylark.



Checklista

Ställer du dessa frågor till leverantören har du ett bra underlag för att bedöma integrationerna:

1. Har systemet ett samlat API, eller har det många separata API:er?
Be gärna om att få se hur en integration faktiskt sätts upp i praktiken.

2. Kan vi hämta all relevant data i en enda fråga, eller måste den sammanställas i efterhand?
Om svaret är "flera anrop + efterbearbetning" ökar komplexiteten snabbt när ni börjar använda systemet.

3. Är API:et fundamentet i systemet, eller är det ett tillägg?
Fråga gärna också: *Finns det funktioner eller data i systemet som vi inte får tillgång till via API:et*? Om svaret på detta är ja, kommer ni förr eller senare behöva bygga tungrodda "workarounds" eller manuella processer.

4. Vad händer när vi behöver något som systemet inte stödjer i dag?
Måste ni skapa "workarounds" (→ kaos), eller kan ni utöka datamodellen själva (→ kontroll)?"

5. Blir nya integrationer enklare eller mer komplexa över tid?
Ha följande i bakhuvudet: Ett bra system gör *nästa integration enklare än den förra*.

6. Hur beroende är vi av dig som leverantör för att göra ändringar?
Om beroendet är stort blir ert tempo lägre.

7. Hur hanterar systemet ändringar i befintliga integrationer?
Små ändringar bör inte kräva stora projekt.

8. Hur säkerställer systemet att det finns en gemensam källa till sanning (single source of truth) i alla integrationer?
Be gärna om ett konkret exempel på hur samma data används konsekvent i flera system. Flera versioner = lägre beslutskvalitet.

9. Hur bidrar systemet till att minska det manuella arbetet i praktiken?
Leverantören bör kunna visa vad som typiskt ersätter Excel, manuella exporter/importer och liknande mellanlösningar.

10. Hur stödjer systemet beslut i realtid, inte bara rapportering i efterhand?
Fråga hur du som ledare faktiskt kan använda data från systemet i operativa beslut. Detta är det ultimata testet på om affärssystemet är värdeskapande.

Om du inte får bra och betryggande svar på dessa frågor riskerar du att investera i ett ERP-system som löser gårdagens behov, inte dagens och morgondagens.

FAQ om integrationer

Vad är egentligen GraphQL?
– GraphQL utvecklades av Meta 2012, lanserades som öppen källkod 2015 och används i dag av företag som GitHub, Shopify och Netflix. Enligt IBM och Gartner pekar flera analyser på en stark tillväxt för GraphQL inom enterprise-arkitektur och modern API-design.

Vad är skillnaden mellan ett traditionellt API och GraphQL i ett ERP-system?
– Ett traditionellt REST-API kräver separata anrop för varje datamängd, medan GraphQL gör att du kan hämta all relevant data i en enda fråga via en enda slutpunkt. Detta minskar komplexiteten och ger snabbare svarstider.

Vad betyder API-first för oss som företag?
– API-first betyder att hela ERP-systemets datamodell är byggd för att vara tillgänglig via API från dag ett. Du slipper upptäcka att data som du behöver inte är tillgänglig för integrationer.

Vad är en flexibel datamodell (DME), och varför är det viktigt?
– Med DME (Data Model Extension) kan du lägga till egna fält och datastrukturer i ERP-systemet utan att bygga lösningar på utsidan. Allt du lägger till blir automatiskt tillgängligt i API:et.

Hur vet vi om vårt ERP-system är redo för AI?
– Ett ERP-system är redo för AI när alla relevanta data är tillgängliga via ett strukturerat API utan manuella mellanled. Ju mer komplett API:et är, desto enklare är det att koppla på AI-verktyg.

Som ledare i dag befinner du dig i ett landskap där tempot ökar, komplexiteten växer och kraven på beslut blir allt högre.

Då är det lätt att se ERP-systemet främst som ett kontrollverktyg – något som måste vara på plats för att säkra compliance. Men min erfarenhet är att de företag som verkligen lyckas ser på det på ett lite annat sätt.

De använder sitt ERP-system som en källa till flöde, fart och bättre beslut.

Skillnaden ligger sällan i ambitionsnivån, utan i vilka förutsättningar de har för att få sina system att samspela i praktiken.

Därför hoppas jag att den här guiden har gett dig en tydligare bild av vad som skiljer ett ERP-system som bromsar från ett som faktiskt driver verksamheten framåt. I en vardag där osäkerheten är stor och besluten många, är det just detta som kan utgöra skillnaden.

Jag önskar dig stort lycka till med de val du står inför.

Øyvind Larsen
Managing Director, Visma Software Nordic