Whitepaper

Systeemintegratie: Het fundament onder uw digitale strategie

AUTODESK Platinum Partner

CADAC GROUP

Inhoudsopgave

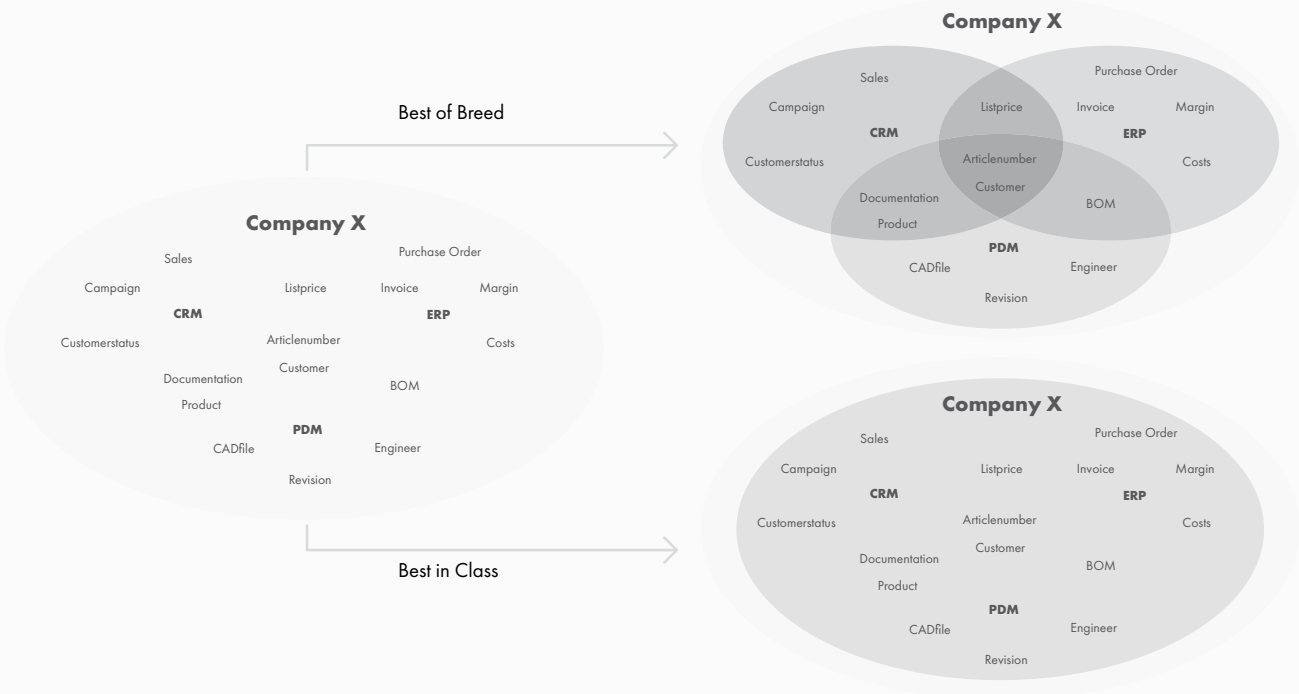
Over Cadac Group

In de maakindustrie, de bouw en bij de overheid is digitalisering in volle gang. De manier waarop u ontwerpt, maakt en gebruikt verandert radicaal. Onze experts helpen u deze digitale transformatie te omarmen.

Dat doen we door u optimaal gebruik te laten maken van de mogelijkheden van intelligent digitaal ontwerp, waarbij modelgebaseerd werken het uitgangspunt vormt. Dat doen we ook door via 'design for manufacturing' en 'design to build' de realisatie van uw ontwerpen veel effectiever te laten verlopen. En dat doen we door slim gebruik te maken van de data die uw ontwerpen genereren.

We stellen u in staat een nieuwe fase van groei in te gaan. Zodat u klaar bent voor de toekomst. Wat die ook brengt.

Cadac. Enabling digital.



Links: *bedrijfsinformatie.*
Rechts boven: *beheer van bedrijf informatie in verschillende applicaties.*
Rechts onder: *beheer van bedrijf informatie in één applicatie.*

1. Inleiding

Digitalisering van bedrijfsprocessen of 'digitaal transformeren' heeft natuurlijk primair te maken met strategie. Dit document gaat niet over de strategie die aanleiding geeft tot het digitaliseren van uw bedrijfsprocessen, maar over een belangrijke bouwsteen in de concrete invulling van uw digitale strategie, namelijk het integreren van uw applicatielandschap, ofwel systeemintegratie.

Het begrip systeemintegratie zal in dit document veelvuldig gebruikt worden en behoeft enige context. Dit document gaat niet over systeemintegratie binnen de industriële automatisering (PLC, SCADA, DCS en MES), maar over het integreren van bedrijfstoepassingen zoals ERP, CRM, EDM, PDM etc. Met andere woorden, dit document beschrijft hoe 'business-data', vaak geproduceerd door mensen, tussen systemen uitgewisseld kan worden. Dit document behandelt niet het ophalen, integreren en visualiseren van 'device data', afkomstig van sensoren of PLC's. Dit laatste is namelijk typisch het domein van de industriële automatiseerders.

De aanleiding om bedrijfsapplicaties met elkaar te integreren is heel simpel: mensen zijn gemakzuchtig en kunnen fouten maken. Om die reden is het beter om informatie ingevoerd in één systeem (bijvoorbeeld het adres van een klant in CRM) direct automatisch te tonen in een ander systeem (bijvoorbeeld de financiële administratie) in plaats van deze opnieuw in te voeren. Vaak wordt er door IT'ers gesproken over een 'Single Source of Truth', een principe waarbij een informatie-element (bijvoorbeeld een adres) maar één keer mag worden opgeslagen om te voorkomen dat er twee waarheden ontstaan en er te veel opslagruimte in beslag wordt genomen. Een mooi principe, maar ook utopisch en praktisch lastig uit te voeren. Allereerst kost opslagruimte bijna niets, dat argument is dus niet meer valide. Probeer als bedrijf maar eens te kiezen voor bedrijfstoepassingen die gebruikmaken van volledig gescheiden informatiedomeinen, dat is onmogelijk. Of het nu gaat om bedrijfsgegevens, omzetgegevens of artikelgegevens, er zal altijd sprake zijn van overlappende informatie binnen de verschillende toepassingen. Daarom zal er altijd behoefte zijn aan het integreren van toepassingen, zodat de informatie op meerdere plekken zichtbaar is maar op één plek beheerd wordt.

Figuur 1: Best-of-Breed vs. Best-in-Class

Er zijn mensen die zeggen dat dit voorkomen kan worden door voor één toepassing te kiezen die het volledige bedrijfsproces ondersteunt, zodat de behoefte om systemen te integreren niet meer bestaat. In IT-jargon heet dat een 'Best-in-Class'-benadering, die haaks staat op de 'Best-of-Breed'-benadering die pleit voor een het ondersteunen van bedrijfsprocessen door middel van een veelvoud aan toepassingen. Voor beide benaderingen is veel te zeggen, maar in de afgelopen jaren tekent zich een duidelijke ontwikkeling in de richting van de 'Best-of-Breed'-benadering. Dit komt met name doordat bedrijfsprocessen steeds sneller veranderen, digitaal worden en het ondersteunen van deze verandering met één monolithisch systeem complex, risicovol en duur is. Ook is de afhankelijkheid van één leverancier, de zogenaamde 'vendor lock-in', een sterk argument om te kiezen voor een heterogeen applicatielandschap.

In dit document wordt uitgelegd hoe systeemintegratie werkt en welke keuzes een bedrijf kan maken om een solide fundament onder hun applicatielandschap te kunnen leggen.

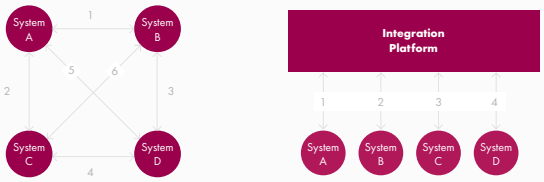
2. Manieren om systemen te integreren

Er zijn allerlei manieren om systemen met elkaar te integreren. In essentie zijn er twee strategieën om systemen met elkaar te integreren:

1. Bilaterale (ofwel point-to-point) koppelsystemen of;
2. Koppel systemen via een platform.

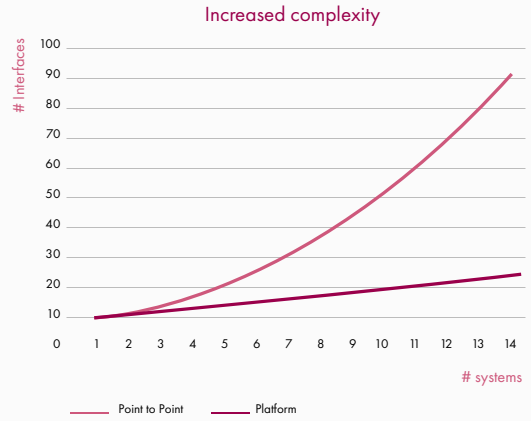
Om te begrijpen wat het essentiële verschil is tussen beide integratiemethoden is een metafoor wellicht handig. Stel uw gezin bestaat uit drie kinderen. U besluit om te gaan verhuizen en u wilt dat met uw gezinsleden bespreken. U kunt ervoor kiezen om een individueel gesprek te hebben met ieder kind zodat u kunt inspelen op de specifieke behoefte van ieder kind. Dit is analoog met de bilaterale strategie; u levert dus voor ieder kind maatwerk in uw communicatie, maar de inspanning in tijd is ook een stuk groter en u heeft misschien niet ieder kind dezelfde informatie gegeven.

Een alternatieve benadering is om het gezin bij elkaar te roepen en het verhaal één keer te doen. Ieder gezinslid hoort dan op hetzelfde moment het commentaar van elkander. Dit is analoog met de platformstrategie; uw boodschap is gelijk voor iedereen en iedereen hoort de feedback van ieder gezinslid op hetzelfde moment. U kunt zich voorstellen dat naarmate het gezin groeit, de tweede aanpak meer voor de hand ligt Zeker als u ervan uitgaat dat de kinderen onderling ook met elkaar praten en een inconsistente boodschap tot allerlei misverstanden kan leiden. Zo is het ook met systemen. Hoe meer systemen, hoe minder effectief de bilaterale benadering is.



Figuur 2: aantal koppelingen bij een bilaterale integratie versus een integratieplatform.

Bovenstaand figuur vergelijkt het aantal koppelingen van een bilaterale integratie met een integratieplatform. Links ziet u het aantal bilaterale koppelingen bij een volledige integratie van vier systemen (alle systemen wisselen informatie uit in meerdere richtingen). Rechts ziet u het aantal koppelingen bij een integratie via een platform. Bij vier systemen zijn dat er al twee minder en dat verschil neemt exponentieel toe bij de stijging van het aantal systemen (zie figuur 3).



Figuur 3: exponentiele groei van de complexiteit bij toename van het aantal systemen.

Om beter te begrijpen waarom de complexiteit bij bilaterale (point-to-point) koppelingen veel hoger is dan bij een platform moeten we kijken naar de fundamentele eigenschappen van een koppeling tussen twee systemen. Opnieuw een metafoor op basis van de communicatie tussen twee mensen. Stelt u zich voor dat een blinde Fransman de weg vraagt aan een dove Engelsman. Er doen zich op dat moment een aantal problemen voor:

1. De Engelsman hoort de vraag niet – het communicatieprotocol is niet compatibel.
2. Ook nadat de Fransman zijn vraag op een blaadje schrijft, begrijpt de Engelsman de vraag nog steeds niet, omdat hij geen Frans spreekt – de informatie moet dus eerst vertaald worden voordat de twee elkaar begrijpen.
3. De Fransman krijgt geen, of een voor hem onbegrijpelijke reactie op zijn vraag – er treedt een fout in de communicatie op die netjes afgehandeld moet worden.
4. Het is zeer waarschijnlijk dat beide personen, ook na meerdere pogingen, elkaar niet zullen begrijpen en dat de Fransman verdwaalt – de informatie-uitwisseling is dus niet erg robuust en betrouwbaar.

Een koppeling tussen twee systemen moet dus rekening houden met al deze onderdelen:

- Het **communicatieprotocol**: afhankelijk van het systeem is er een veelvoud aan protocollen die mogelijk gebruikt moeten worden, zoals Directory Polling, FTP, SOAP over HTTP(S), REST over HTTP(S), etc.
- De **vertaalslag**: data uit beide systemen moet verbonden en veelal vertaald worden, zodat de juiste informatie wordt uitgewisseld. Op die manier kan een klant 'customer' heten in het ene systeem en 'debtor' in het andere.
- Betrouwbare **foutafhandeling**: als er iets misgaat in de communicatie tussen twee systemen, bijvoorbeeld doordat één van de twee systemen offline is, dan moet dit bij de juiste personen tijdig kenbaar gemaakt worden, zodat er maatregelen genomen worden.
- De communicatie moet **robuust** zijn: een tijdelijke storing in de communicatie, doordat bijvoorbeeld een datalijn overbelast is, mag niet altijd tot een fout leiden. De koppeling moet bij een probleem automatisch de actie herhalen en pas als het na een aantal keer opnieuw proberen niet lukt, een foutmelding genereren waarop een persoon moet reageren.

Bij iedere bilaterale koppeling moet een programmeur rekening houden met bovenstaande aspecten. Er is een grote kans dat iedere programmeur dit op zijn of haar eigen manier doet, als het al gebeurt. De koppeling wordt hierdoor niet alleen minder betrouwbaar, maar ook duur in de aanschaf en duur in het onderhoud. Bij iedere wijziging moeten programmeurs ingeschakeld worden en moet er kennis van de ene persoon op de andere overgedragen worden.

Omwille van bovenstaande redenen is het beter om te kiezen voor een integratieplatform dat de noodzaak voor een programmeur grotendeels overbodig kan maken. In het volgende hoofdstuk wordt uitgelegd waarom dat zo is.

3. Het integratieplatform uitgelegd

We hebben gezien dat een integratieplatform een goede oplossing biedt voor bedrijven met een groot aantal applicaties welke geïntegreerd moeten worden. Om te begrijpen waarom een integratieplatform een goed idee is, is het belangrijk om de basisprincipes van een dergelijk platform te begrijpen. Om de werking van een integratieplatform eenvoudig uit te leggen moet de werkelijkheid, die helaas zeer complex is, sterk vereenvoudigd worden.

3.1 Wat doet een integratie platform?

Een integratieplatform verbindt systemen zonder dat deze systemen van elkaars bestaan afweten. Het platform verbindt systemen, transporteert informatie tussen systemen en vertaalt deze onderweg zodat systemen met elkaar kunnen praten. Het platform doet dat betrouwbaar en robuust en zorgt in het uiterste geval dat beheerders worden geïnformeerd als er fouten in de communicatie optreden.

Laten we het volgende voorbeeld nemen:

Stap 1: In het CRM-systeem wordt een nieuwe klant aangemaakt.

Stap 2: De klantgegevens moeten in ERP bekend gemaakt worden, zodat de factuur vanuit hier gestuurd kan worden.

Binnen het CRM-systeem heeft een klant een nummer 'CRM ID' en binnen het ERP-systeem heeft de klant een 'Debtor Number' van acht karakters dat altijd begint met de letters van het land, bijvoorbeeld NL.

Beide informatie-elementen zijn specifiek voor het desbetreffende systeem. Om ervoor te zorgen dat het integratieplatform niet afhankelijk is van één van deze systemen, besluiten we het unieke klantnummer te definiëren als 'Customer ID'.

De communicatie tussen beide systemen zal als volgt verlopen:

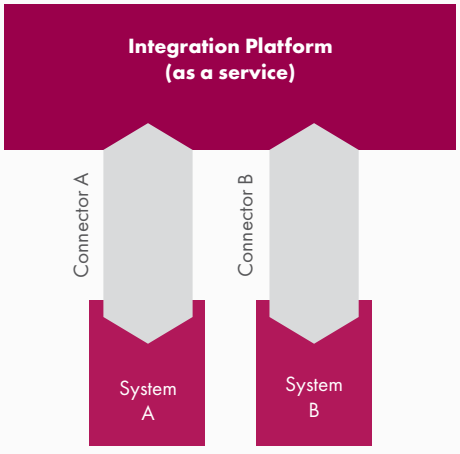
1. In CRM wordt de klant met 'CRM ID = 1234' aangemaakt;
2. Het integratieplatform vertaalt het 'CRM ID' naar 'Customer ID = 1234' en weet dat ERP deze informatie nodig heeft;
3. Het integratieplatform vertaalt 'Customer ID = 1234' naar 'Debtor Number = NL001234' en biedt deze informatie aan het ERP-systeem aan;
4. Het integratieplatform krijgt een bevestiging van het ERP-systeem dat de klant is aangemaakt. Dat is belangrijk, wantanders worden orders van deze klant niet goed verwerkt en zal er nooit een factuur naar de klant gaan.

Stel dat het bedrijf met beide systemen de overstap wil maken naar een nieuw ERP-systeem, dan hoeft het bedrijf alleen maar het nieuwe ERP-systeem aan te sluiten op het integratieplatform. Het CRM-systeem merkt daar niets van. Dat kan doordat de informatie die is uitgewisseld vertaald wordt naar een bedrijfsstandaard, in IT-jargon ookwel 'Master Data Management' genoemd.

3.2 Onderdelen van een integratie platform

Kijken we van grote afstand naar een integratieplatform, dan bestaat dit uit een paar onderdelen:

1. Het platform: de stekkerdoos die ervoor zorgt dat gegevens van de ene naar de andere applicatie worden getransporteerd.
2. De connectoren: de stekkers die aansluiten op de te integreren systemen en het platform.

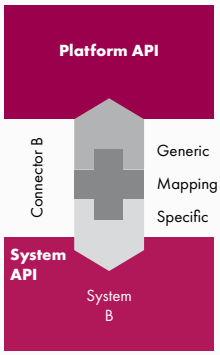


Figuur 4: schematische voorstelling van een integratieplatform.

3.2.1 Connectoren

De connectoren vormen de koppeling tussen een systeem en het platform en zijn specifiek voor een bepaald systeem. Als we verder inzoomen op de connectoren (ook vaak 'adapters' genoemd), dan zien we dat deze bestaan uit een aantal onderdelen die ervoor zorgen dat het onderliggende systeem aangesloten kan worden:

1. Een onderdeel dat specifiek past op een bepaald systeem. Dit onderdeel kan informatie uitwisselen met het systeem door middel van een koppelvlak, ook wel API (Application Programming Interface) genoemd.
2. De vertaling van gegevens uit het systeem met de bedrijfsstandaard, zoals dit is vastgelegd in de 'master data'.
3. De aansluiting op het integratieplatform dat vervolgens weet welke andere systemen geïnteresseerd zijn in de informatie uit de connector.



Figuur 5: onderdelen van een connector.

Door de systeemspecifieke gegevens te vertalen naar bedrijfsgegevens, ontstaan er gegevens in het integratieplatform die niets weten van de onderliggende systemen en dus uitwisselbaarheid garanderen. Verder kan deze bedrijfsspecifieke definitie van gegevens gebruikt worden om te rapporteren over systemen heen.

3.2.2 Informatiestroom

Een belangrijk probleem is nog niet besproken, namelijk: hoe weet het ene systeem dat het andere systeem belangrijke informatie heeft aangemaakt of gewijzigd? Ofwel, op welke manier wordt de stroom van informatie geregeld? Voor de informatiestroom moeten we twee gevallen onderscheiden:

1. Directe/synchrone informatiestroom;
2. Indirecte/asynchrone informatiestroom.

Om het verschil goed te begrijpen beschouwen we de volgende voorbeelden:

- Voorbeeld van een synchrone informatiestroom: Een klant logt in op de website van een bedrijf en wil de orderhistorie uit ERP inzien. Om dit mogelijk te maken, moet de bedrijfswebsite (systeem A) via het integratieplatform met het ERP-systeem (systeem B) gekoppeld worden. Als een gebruiker op een website op een knop drukt om de orderhistorie uit het achterliggende ERP-systeem in te zien, dan verwacht de gebruiker binnen enkele seconden het resultaat in zijn webbrowser te zien.
- Voorbeeld van een asynchrone informatiestroom: Een klant bestelt een reserveonderdeel van een machine via de website. Om dit mogelijk te maken moet de bedrijfswebsite (systeem A), via het integratieplatform met het ERP-systeem (systeem B) gekoppeld worden. Als de bezoeker van de website op de 'bestellen' knop drukt, dan hoeft de inkoper die met het ERP-systeem werkt, niet direct op de hoogte gebracht te worden van deze bestelling, dat mag best een minuut duren.

Om dit verschil in de verwachtingen van eindgebruikers mogelijk te maken, moet het integratieplatform dus beide soorten informatiestromen kunnen ondersteunen. Het platform ondersteunt synchrone communicatie door het mogelijk te maken om vragen vanuit een gebruikerstoepassing direct door te sluizen naar één of meerdere gekoppelde bronsystemen. Moderne integratieplatforms bieden hier ondersteuning voor in de vorm van 'listener API's' die oproepen van systemen opvangen, vertalen naar gekoppelde systemen en direct het antwoord teruggeven.

Asynchrone communicatie wordt mogelijk gemaakt door (batch-)jobs die op gezette tijden of door middel van triggers aan de onderliggende systemen vragen of er nog iets is veranderd. Het uitvoeren van dit soort jobs kost rekenkracht en tijd en is daarom minder snel en efficiënt dan een synchrone communicatie. Het uitvoeren van dit soort jobs kost rekenkracht en tijd en is daarom minder snel en efficiënt dan een synchrone communicatie. Daarentegen, jobs hebben als voordeel dat ze grote hoeveelheden informatie kunnen verwerken. Verder is asynchrone communicatie in de regel veel robuuster dan synchrone communicatie, omdat een integratieplatform over zogenaamde wachtrijen beschikt die informatie vasthouden totdat deze afgeleverd kan worden. Dat is met name het geval wanneer een gekoppeld systeem offline is, of het op een bepaald moment te druk heeft om informatie van het integratieplatform te ontvangen. Het integratieplatform zal de informatie blijven aanbieden en een beheerder van het platform notificeren zodra het na een aantal keer proberen nog steeds niet is gelukt. Deze robuustheid bestaat niet voor synchrone communicatie; als het gekoppelde systeem niet antwoordt dan zal de gebruiker een foutmelding krijgen en dat is natuurlijk niet erg gebruikersvriendelijk. Bij het selecteren en inrichten van een integratieplatform moet dus met dit soort aspecten rekening gehouden worden.

3.2.3 Master Data Management

Zoals in bovenstaand voorbeeld wordt uitgelegd is het van groot belang om in het integratieplatform de informatiesysteem neutraal uit te wisselen, zodat een 'vendor lock-in' kan worden voorkomen. In de praktijk wordt ook vaak het bronsysteem als leverancier van de 'master data' aangewezen. Stel dat een nieuwe klant altijd in CRM aangemaakt wordt, dan kan ervoor gekozen worden om de CRM-sleutel als unieke sleutel voor klantgegevens aan te wijzen. Het nadeel van deze pragmatische werkwijze is dat er een afhankelijkheid naar het CRM-systeem wordt ingebouwd, waardoor het vervangen van dit systeem minder eenvoudig zal worden. Alleen door de 'master data' goed te definiëren is het mogelijk om de vertaalslag in de connectoren goed te regelen. Door gebruik te maken van goed gedefinieerde ''master data' wordt het ook makkelijker om eenduidige informatie te onttrekken aan alle systemen. In plaats van het bouwen van rapporten op individuele systemen kan ervoor gekozen worden om een zogenaamd 'datawarehouse' te bouwen dat de belangrijkste data van een bedrijf bevat. Het hebben van een goede definitie van de 'master data' is een belangrijke randvoorwaarde voor het bouwen van een 'datawarehouse'.

4. Conclusie

In dit document wordt uitgelegd waarom het een goed idee is om systemen te integreren door middel van een integratieplatform, met name als er meer dan drie applicaties met elkaar gekoppeld moeten worden. Koppelingen tussen systemen via een integratieplatform zijn niet alleen betrouwbaarder, eenvoudiger te bouwen en te onderhouden dan bilaterale koppelingen, maar voorkomen ook dat er sprake is van een 'vendor lock-in', waardoor bedrijven sneller kunnen blijven inspelen op de steeds veranderende behoefte aan digitalisering.

De keuze voor het juiste integratieplatform wordt door een groot aantal factoren bepaald. De belangrijkste onderscheidende factor is het IT-kennisniveau dat een bedrijf in huis heeft. Er zijn zogenaamde 'No Code'-platforms die beloven dat integraties door iedereen bij elkaar geklikt kunnen worden. Dat is absoluut een verkeerde voorstelling van zaken. Ook integratieplatforms vereisen de nodige IT-kennis en analytische vaardigheden van de mensen die het platform beheren. Kennis van programmeurs zal zeker in complexe omgevingen onontbeerlijk zijn. Door gebruik te maken van een integratieplatform is het eenvoudiger om de controle te behouden over het applicatielandschap, omdat het maatwerk tot een absoluut minimum wordt beperkt. Ook als bedrijven geen programmeurs in dienst hebben, is dit een goede oplossing. Aangezien (goede) programmeurs nog altijd schaars zijn, is het zaak dat bedrijven zich niet te afhankelijk maken van individuen of leveranciers van programmeerdiensten. Een integratieplatform kan hieraan een belangrijke bijdrage leveren. Het belangrijkste voordeel van een integratieplatform blijft echter dat bedrijven sneller en effectiever invulling kunnen geven aan hun digitale ambitie.

Ontdek meer over systeemintegratie →