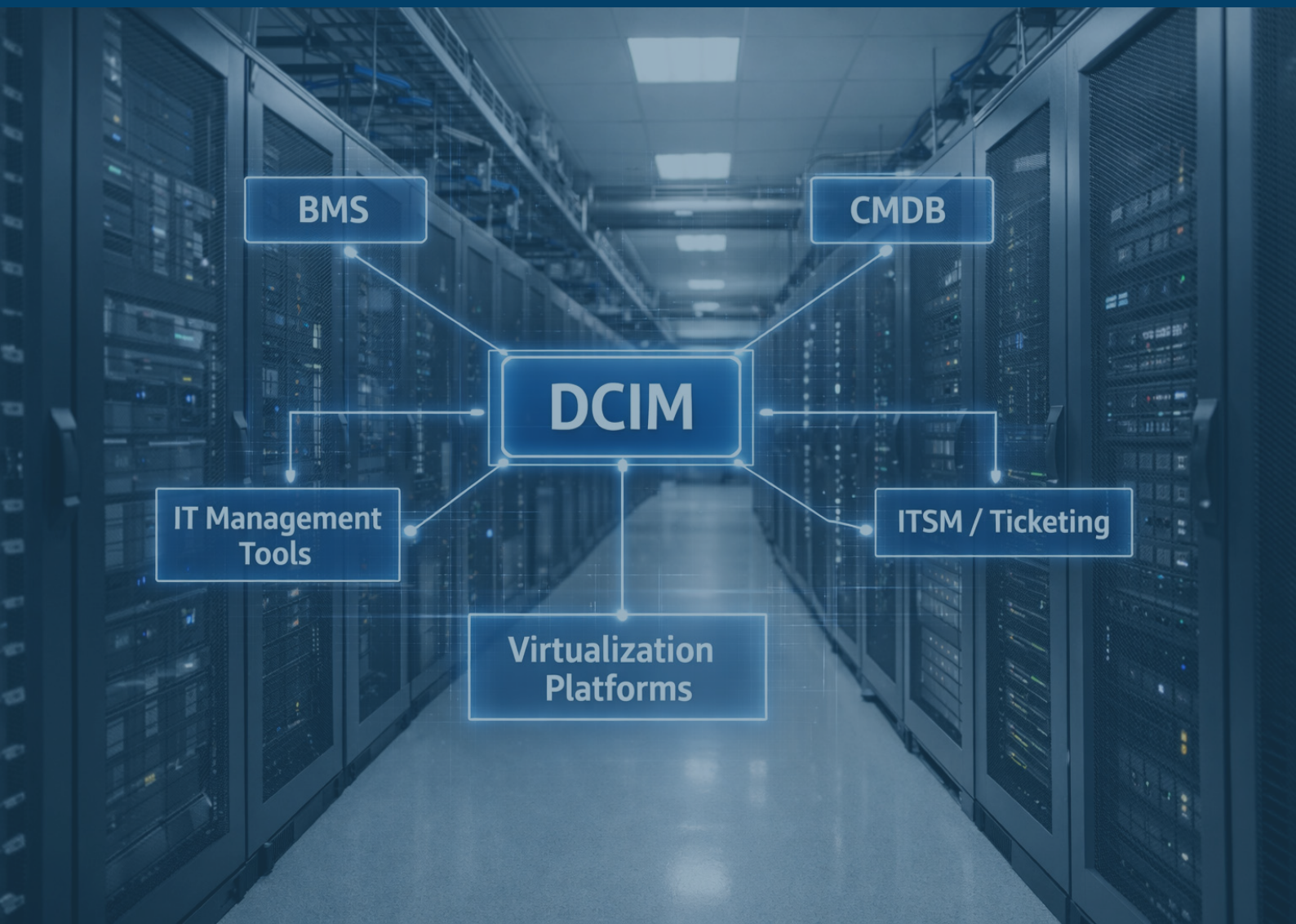
Bjumper

Guia prática de integrações em DCiM

Como conectar o seu DCiM com o ecossistema do Data Center sem morrer na tentativa



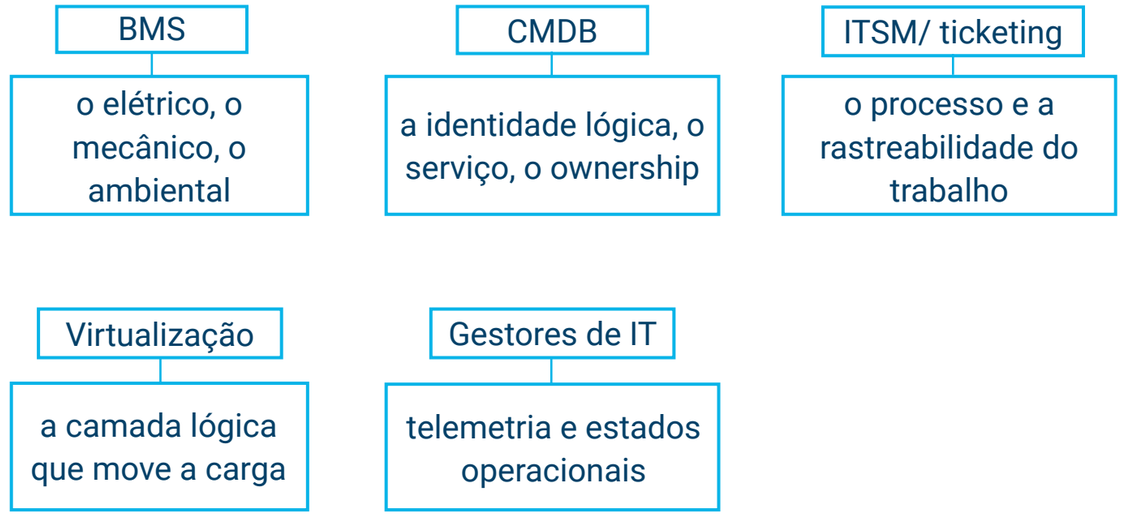
Introdução

Um DCiM “sozinho” já traz valor, mas um DCiM conectado é quando começa a se tornar uma ferramenta de automação real para o seu Data Center: passa a ser uma peça viva que **recebe sinais, coloca-os em contexto e os transforma em ação.**

A integração em DCiM não é sobre “trazer dados por trazer”. É sobre responder perguntas operacionais:

- O que está realmente acontecendo com energia e clima, e como isso impacta meus racks?
- Onde está a fonte da verdade do inventário e quem manda em cada campo?
- Como reduzo dupla carga de trabalho e erros humanos?
- Como cruzo o mundo lógico (VMs/serviços) com o mundo físico (racks/hosts)?
- Onde tenho minha informação de serviço e como a coloco à disposição do que acontece na infraestrutura?

O Data Center real não vive em um único sistema. Vive distribuído entre:



E todos eles respondem a um mesmo objetivo, entregar o serviço em tempo e com eficácia para a continuidade do negócio.

Depois de muitas reuniões e integrações com clientes, na Bjumper decidimos fazer este guia para ajudar você, e nos ajudar, a desenhar e executar integrações de forma prática, com um enfoque pensado no que aporta, no outcome (o impacto real).

Por isso é importante o que integrar, por quê, como fazer bem e o que evitar. Porque o problema não é que existam muitos sistemas onde os dados se encontram. O problema é quando eles não se falam, aí você duplica tarefas, a coerência se rompe, aparecem “verdades paralelas” e a operação se torna reativa.

De fato, este guia em nível conceitual poderia servir para qualquer integração entre sistemas, não envolvendo o DCiM ou até mesmo sistemas fora do ambiente do Data Center.

Os Objetivos

Embora soe óbvio, definir os objetivos que as integrações entre sistemas buscam é primordial, alinha expectativas entre departamentos e suaviza fricções, para não dizer que obriga você a fazer as perguntas adequadas.

Objetivo 1: Integridade do dado (o primeiro e este não pode ser pulado)

Sem identificadores consistentes, regras claras e qualidade mínima, qualquer integração se converte em uma fábrica de inconsistências. **O DCiM não "conserta" o dado: ele o amplifica.**

Portanto, a escolha dos identificadores deve ser correta (têm que ser únicos e imutáveis). Se não forem, qualquer integração acaba derivando em duplicidades, colisões ou dependências frágeis difíceis de manter.

O tipo de integração determina a integridade do dado. Não é a mesma coisa uma integração unidirecional que uma bidirecional. A primeira é mais simples de governar; a segunda introduz muito mais complexidade e obriga a definir regras claras de sincronização e resolução de conflitos.

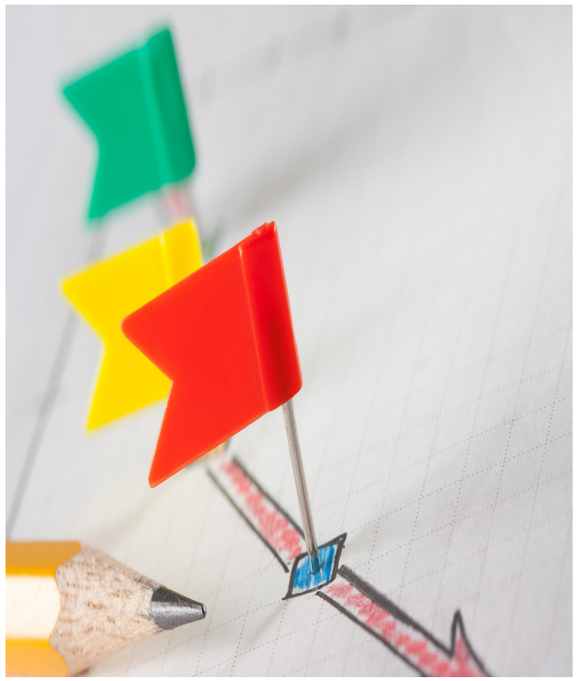
Se você traz informação de outros sistemas para o DCiM, é fundamental entender que a governança desse dado não é do DCiM, mas sim da origem. Se a origem muda, deveria mudar no DCiM, não o contrário. Em integrações bidirecionais isso é especialmente crítico: é preciso definir ownership por dado e qual sistema tem prioridade quando ambos modificam a mesma informação ao mesmo tempo, ou seja, a fonte da verdade.

Bjumper

Objetivo 2: Eficiência operacional

Pensar de forma conjunta e conforme os seus processos, esta integração como responderia a como nos torna mais eficientes na operação?

- Levaremos menos tempo em alguma tarefa?
- Não teremos dupla carga de trabalho ao incluir o mesmo dado em 2 lugares?
- Permitirá que tenhamos menos erros?
- Confierei mais no dado que o sistema está me dando e não farei uma dupla verificação onsite ou com outro sistema?
- Poderei medir a melhoria?



Antes de ativar qualquer integração, defina como você vai saber se funcionou. Pode ser algo tão simples quanto medir quanto tempo você dedicava a uma tarefa antes e quanto dedica depois, ou quantos erros de inventário detectava por mês.

Sem uma referência mensurável, a eficiência é uma percepção, não um fato, e quando perguntarem se a integração valeu a pena, você precisa responder com dados, não com sensações.

Objetivo 3: Contexto (o que diferencia o DCiM)

Se vamos conectar um sistema com o DCiM é preciso definir muito bem o que ele vai aportar ao DCiM para dar mais contexto e buscar a automação e o que o DCiM aportaria ao sistema para enriquecer seus dados.

O DCiM não compete com sistemas tão válidos e próprios do ambiente do Data Center como o BMS, CMDB, ITSM, Virtualização, Sistemas AIMs, networking etc, mas também não devemos confundi-los com o conceito próprio do DCiM.

Portanto, a pergunta mágica é: O que o DCiM contextualiza?

Por exemplo:

- **Do BMS,** a energia e o clima enriquecerão o conhecimento por sala, fila, rack e serão a base para relatórios de consumo por linha de negócio, correlação de incidentes e como se antecipar, etc.
- **Da CMDB,** o proprietário de um equipamento, para comunicar o estado do ativo à pessoa adequada, inclusive em momentos de manutenção preventiva, assim como o DCiM pode aportar confiabilidade aos dados relacionados com a camada física.

Da mesma forma, não podemos esquecer que o DCiM já dispõe de muito valor que também vai compartilhar com outros sistemas, por exemplo:

Bjumper

DCiM → CMDB

O DCiM devolve à CMDB
a realidade física do ativo. Localização exata (site, sala, fila, rack, U), estado real de ocupação, conectividade física (portas, cabeamento), potência nominal e consumida. Quando a CMDB recebe essa informação do DCIM, deixa de depender de alguém atualizar manualmente um campo de localização que na maioria dos casos está desatualizado.

DCiM → ITSM

O DCiM aporta ao ITSM:
contexto para a execução e evidência para o fechamento. Quando um técnico recebe um ticket de instalação, o DCiM diz exatamente onde há espaço, potência disponível e conectividade livre.

No fechamento, o DCiM pode confirmar que o trabalho foi executado (o equipamento está registrado, localizado e conectado), o que transforma o fechamento do ticket em algo baseado em evidência, não na palavra de alguém.

Além disso, o DCiM pode ser quem dispara tickets automaticamente: uma manutenção preventiva programada, um alarme de capacidade no limite ou um limite ambiental superado podem gerar um ticket no ITSM sem intervenção humana.

DCiM → Virtualização

O DCiM aporta à plataforma de virtualização algo que ela não tem visibilidade:
o contexto físico do host. Em que rack está? Que alimentação tem? Compartilha circuito elétrico com outros hosts do mesmo cluster?

Objetivo 4: Base para automação (a médio prazo)

Para otimizar os recursos do Data Center, minimizar riscos. A automação não chega por “colocar IA”. Chega quando:

- o dado é confiável,
- os processos estão claros, talvez o ponto mais crítico para a operação de um Data Center.
- a tecnologia/integração sustenta o processo sem fricção.

Conceitos-chave: “Monitorizar” vs “Integrar”

Queremos enfatizar a diferença entre monitorizar versus integrar, para isso vamos defini-las:

Monitorizar

Monitorizar em um DCiM é capturar dados operacionais diretamente do hardware ou de sistemas OT (por exemplo CRACs/UPS/PDUs/sensores), normalmente através de protocolos padrão e com uma frequência alta (minutos ou até mais frequente), para refletir “o que está acontecendo agora” na infraestrutura.

O que traz
medidas e estados (kW/kWh, V/A, temperatura, humidade, estado UPS/gerador, alarmes básicos…).

Fluxo típico:
o DCiM recolhe os dados diretamente do HW, normalmente, salvo casos muito específicos, será unidirecional, já que o DCIM não vai controlar o HW, não é um sistema de controlo.

Para que serve
operação e controlo em tempo quase real: capacidade, risco, e colocar alarmes com contexto, que estes sim podem ir ao DCiM em tempo real, mas também não é um NOC.

Integrar

Integrar em um DCiM é conectar o DCiM com outros sistemas de gestão (CMDB, ITSM/ticketing, virtualização, gestores IT, AIM, BMS…) mediante API contra API (ou outros mecanismos de integração), para sincronizar e complementar informação e, sobretudo, conectar processos entre equipas.

O que traz
identidade e atributos de ativos (CI, serviço, owner, estado lógico, hostname/IP), processos (tickets/ordens/estados), relações lógicas (VM ⟷ host), etc

Fluxo típico:
ode ser unidirecional ou bidirecional, segundo a governação do dado.

Para que serve
reduzir dupla carga, manter coerência entre sistemas e habilitar rastreabilidade end-to-end para começar com a automação.

Ou seja, a monitorização é uma capacidade que o DCiM possui onde procura extrair dados de equipamentos eletromecânicos ou de IT que são capazes, mediante protocolos de comunicação, de oferecer telemetria, e a integração é ir sistema contra sistema e pode ter várias metodologias segundo as capacidades dos sistemas ou às vezes até protocolos internos das empresas: uma simples troca de ficheiros, chamadas concretas à base de dados com queries ou ir API contra API.
Do nosso ponto de vista, aceder diretamente à base de dados de outro sistema é a opção mais frágil e perigosa. Na medida do possível, é sempre melhor fazê-lo via API.

Princípios de ouro (antes de falar de tecnologia)

1. “Fonte da verdade” por campo

Para cada dado importante define quem manda, isto, que às vezes pode parecer lógico, torna-se momento de discussão na maioria das empresas, porque por trás dos dados estão as pessoas, que tendem a apropriar-se da sua autoria.

A fonte da verdade é que o dado que manda, quando temos diferentes plataformas, é aquele que se encontra mais próximo da origem ou do ativo.

Primeiro, se há dados em plataformas que recolhem automaticamente, veja-se em cablagem inteligente se uma porta está ocupada ou não, não há discussão, mas o tema surge quando existem dois sistemas onde os dados são introduzidos manualmente, a pergunta nesse caso é em qual deles confias mais para esse dado.

Para combater isto, o melhor é fazer testes, pequenas auditorias de ativos onde se possa determinar onde está realmente o dado de qualidade.

Estes casos são os minoritários, há muitos exemplos onde está claro como:

Localização física (rack/U/sala)	Manda DCiM
CI / Serviço / Owner / Estado lógico	Manda CMDB ou ITSM
Métricas elétricas e ambientais	Manda BMS ou sistema de monitorização
Relação VM ⟷ Host	Manda Virtualização

O importante é proteger a qualidade do dado, porque lembrem-se: **o DCiM não “corrige” o dado: amplifica-o.**

2. Identidade inequívoca do ativo

Toda integração precisa de um “match” robusto, e para isso entre os diferentes sistemas deve sempre existir um campo dos ativos a integrar que seja único, inequívoco e consensual.

A regra prática diz-nos que os mais utilizados são:

1.- Serial Number (ideal)
2.- ID único (CI / tag / asset ID)
3.- nome único governado (com norma estrita), o menos recomendado.

Se um ativo não pode ser identificado de forma inequívoca, não o integres “a meio”.
A única coisa que vais criar é ruído, cria a regra de identificação única, põe o processo em marcha e quando isto estiver a 100% então integramos.

3. Integrar “apenas dados de valor”

Com as integrações entre sistemas às vezes pode acontecer como quando abres 20 abas no navegador para ter tudo à mão “para o caso de precisar”, e quando vais procurar algo só cria confusão e às vezes até erros.

Integrar todos os dados disponíveis é:

- caro
- lento
- frágil
- difícil de manter

Antes de decidir que dados vamos integrar, volta a ler os objetivos, isso vai ajudar-te a descartar.

Além disso, como tudo na vida, ou quase tudo, poderás integrar mais dados no futuro. Isto não é estático, é dinâmico conforme as necessidades de cada momento.

Também é importante destacar que a frequência na integração é tão importante quanto os dados.

O tempo de amostragem nas integrações não deve ser igual para todos. Há dados onde é necessário ver mudanças quase instantaneamente e outros onde isso não é necessário.

Por exemplo, a integração com VMware, onde a virtualização é muito volátil, requer um nível maior de consulta ou amostragem do que uma CMDB.

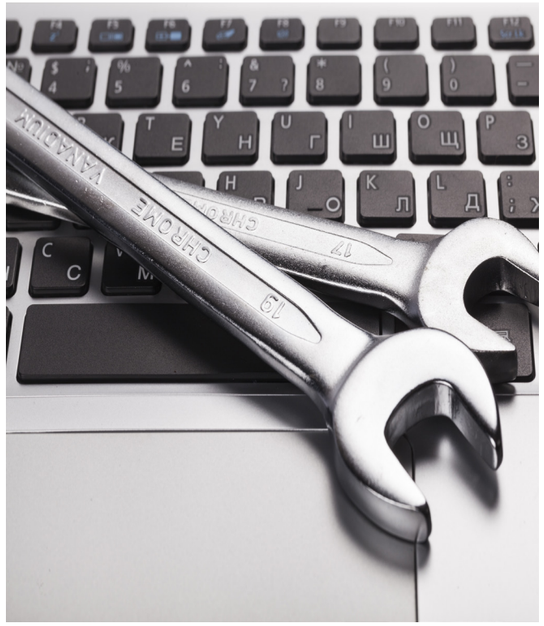
Bjumper

4. Log e exceções como parte do desenho

A integração nunca está “perfeita” no dia 1. Nós aconselhamos desenhar um circuito de melhoria:

- o que não está mapeado ⟶ log
- Monitorizar a saúde da integração tal como fazemos com um serviço
- revisão periódica ⟶ correção de dados / regras
- objetivo: log vazio ou controlado

5. Manutenção



Os sistemas migram de on-premise para cloud, surgem novas versões, muda a forma como os utilizas, por isso não podemos ignorar que as integrações devem ser mantidas ao longo do tempo, adaptadas e, em algumas ocasiões, repensadas e refeitas.

Nestes casos, não se trata de dizer que foram feitas mal na origem, simplesmente não podemos pensar que são eternas e que não requerem a nossa atenção e suporte.

O que fazer e o que não fazer

Vamos a um ponto prático, isto é um guia!

O que fazer

- Definir casos de uso mensuráveis.
- Definir fonte da verdade por campo.
- Definir identificador único e regras de matching.
- Começar por um piloto (âmbito controlado onde confies nos teus dados).
- Mapear poucos campos, mas bem.
- Estabelecer frequência de mapeamento sensata (segundo o tipo de dado).
- Desenhar logs, alertas e um “processo de limpeza”.
- Documentar mapeamentos (mesmo que seja numa tabela simples), assim poderemos fazer acompanhamento.
- Assegurar segurança: credenciais, permissões mínimas, TLS ou rastreabilidade.
- Preparar uma “política de mudanças”: quando mudar a API/versão, o que acontece?

O que NÃO fazer

- Integrar sem governação do dado.
- Integrar sem plano de rollback.
- Usar nomes não únicos ou inventar IDs posteriormente.
- Sobrescrever dados sem regras claras de prioridade.
- Fazer bidirecional “porque soa bem”.
- Integrar com frequências que não acrescentam nada ao modelo (minuto a minuto onde não se aplica).
- Depender de uma pessoa que “sabe como funciona” sem documentação.
- creditar que “se há integração, já há automação”.

Bjumper

A tabela: Fonte da verdade por campo

A seguir apresentamos uma tabela que não tem a verdade absoluta; como dissemos antes, cada empresa tem a sua organização, processos e políticas, mas pode servir como base para uma discussão.

Campo/dato	Sistema “dono” (source of truth)	Sistema(s) que o consomem	Regra de atualização	Frequência	O que acontece em caso de conflito?	Validações mínimas
Identificador único (Serial/ CI/Tag)	CMDB/ DCiM (según política)	Todos	Nunca se sobrescreve sem controlo	N/A	Identificar claramente por caso	Único, não nulo, formato padrão
Localização física (site/ sala/rack/U)	DCiM	CMDB/ ITSM	DCiM manda	Diária / evento	Identificar claramente por caso	Sem duplicados, consistência rack/U
Serviço / Aplicação	CMDB	DCiM/ ITSM	CMBD manda	Diária	Identificar claramente por caso	Necessário em ativos críticos
Owner / Responsável	CMDB/ITSM	DCiM	CMBD/ITSM manda	Diária	Identificar claramente por caso	Campo normalizado
Estado operativo físico	DCiM/BMS	ITSM	É reportado, não se “inven-ta	5-15 min	Identificar claramente por caso	Valores válidos / limiares
Energía (kW/ kWh)	BMS/ Medición	DCiM	BMS manda	5-15 min	Identificar claramente por caso	Unidades, intervalos
Temperatura/ Humedad	BMS/OT	DCiM	BMS manda	5-15 min	Identificar claramente por caso	Sensor mapeado para localização
Ticket/ Workorder	ITSM	DCiM	ITSM cria, DCIM executa/ atualiza	Por evento	IIdentificar claramente por caso	Estados alinhados
VM ↔ Host Rack	Virtualización +DCiM	DCiM/ CMDB	Cada um manda na sua camada	1 h	Identificar claramente por caso	Host físico mapeado

INTEGRAÇÕES MAIS COMUNS

DCiM ⟷ CMDB (inventário físico-lógico com regras)

Para que serve

- Evitar dupla carga de dados.
- Unificar a rastreabilidade do ativo: unir a camada física com a lógica.
- Melhorar auditoria e compliance a que se enfrentam as empresas que incluem o Data Center.
- Preparar o terreno para automatização de processos.

Desenho recomendado

- Bidirecional apenas se houver clareza da fonte da verdade por campo.
- Caso contrário, começar unidirecional e amadurecer.

O que costuma viver em cada lado

- No DCiM: localização física, potência nominal, conectividade física, rack/U, sala.
- Na CMDB: CI, serviço, owner, criticidade, estado lógico, hostname, IP.

Regras práticas de matching

- Serial ou ID único primeiro.
- Se houver colisões bloqueia-se o update e envia-se para log.

Erros típicos

- “A CMDB não está limpa, mas integremos na mesma.”.
- Não limitar o âmbito (entram ambientes não confiáveis e envenena-se o dado).
- Fazer bidirecional sem regras e permitir que ambos os sistemas reescrevam.

Bjumper

DCiM + ITSM/Ticketing (processo e rastreabilidade)

Para que serve

- Conectar a solicitação com a execução e o fecho do projeto.
- Incorporar os solicitantes nos fluxos do Data Center.
- Reduzir tempos de coordenação IT/Ops/Facilities.

Padrões habituais

- Ticket cria projeto/ordem de trabalho no DCiM.
- DCiM devolve estado e fecho ao ticket.
- Disparadores por evento (aprovado, em execução, finalizado).

Dicas práticas

- Definir “tipo de ticket” que dispara trabalho (change / request / incidente).
- Definir campos mínimos obrigatórios nos formulários; se não, o ticket é devolvido.
- Definir quando um trabalho se considera “fechado” (evidência).

Erros típicos

- Integrar sem desenho de processo (apenas “sincronizar estados”).
- Não alinhar taxonomia: estados do ITSM ≠ estados do DCiM.
- Não definir papéis (quem aprova, quem executa, quem v valida).

DCiM + Virtualização (do lógico ao físico)

Para que serve

- Visibilidade de hosts, clusters, VMs, recursos.
- Relação serviços/VMs com ativos físicos.
- Melhor planeamento de capacidade (para além do rack).

Frequência recomendada

- Cada 1 hora costuma ser suficiente na maioria dos casos.
- Se quiser “near real-time”, é preciso justificar o uso (e assumir o custo).

Boas práticas

- Começar com inventário + relações (host/cluster/VM).
- Adicionar consumo/recursos depois.
- Validar reconciliação: uma VM “vive” num host que deve existir fisicamente.

Erros típicos

- Querer “mapear tudo” (etiquetas, políticas, redes complexas) desde o dia 1.
- Não ter um mapeamento sólido host ↔ servidor físico.
- Não acordar o que se considera “fonte da verdade” do estado (ligado/desligado).

DCiM + Gestores IT (métricas IT operativas)

Para que serve

- Trazer telemetria IT relevante para o contexto físico.
- Detetar riscos operacionais (temperatura, consumo, falhas).
- Apoiar capacity planning com sinais reais.

Frequência recomendada

- Diária para inventário/estado base.
- Mais frequente apenas se houver caso claro (alerting operacional).

Boas práticas

- Definir mínimo viável: CPU/mem/consumo/temp/firmware e identificação.
- Gerir exceções (IPs duplicados, mudanças de hostname, etc.)
- Correlacionar com DCiM por serial ou ID robusto.

Errores típicos

- Integrar sem identificador fiável (IP muda, hostname muda).
- Transformar o DCiM num NMS (não é o objetivo).
- Trazer métricas excessivas sem uso real.

A despedida (a pergunta que resume tudo)

Como saber se o fizeste bem?

Se amanhã desligarem a integração, a operação piora de forma mensurável?

Se a resposta for “sim”, fizeste bem: integraste valor.
Se a resposta for “mais ou menos…”, provavelmente integraste “dados”.