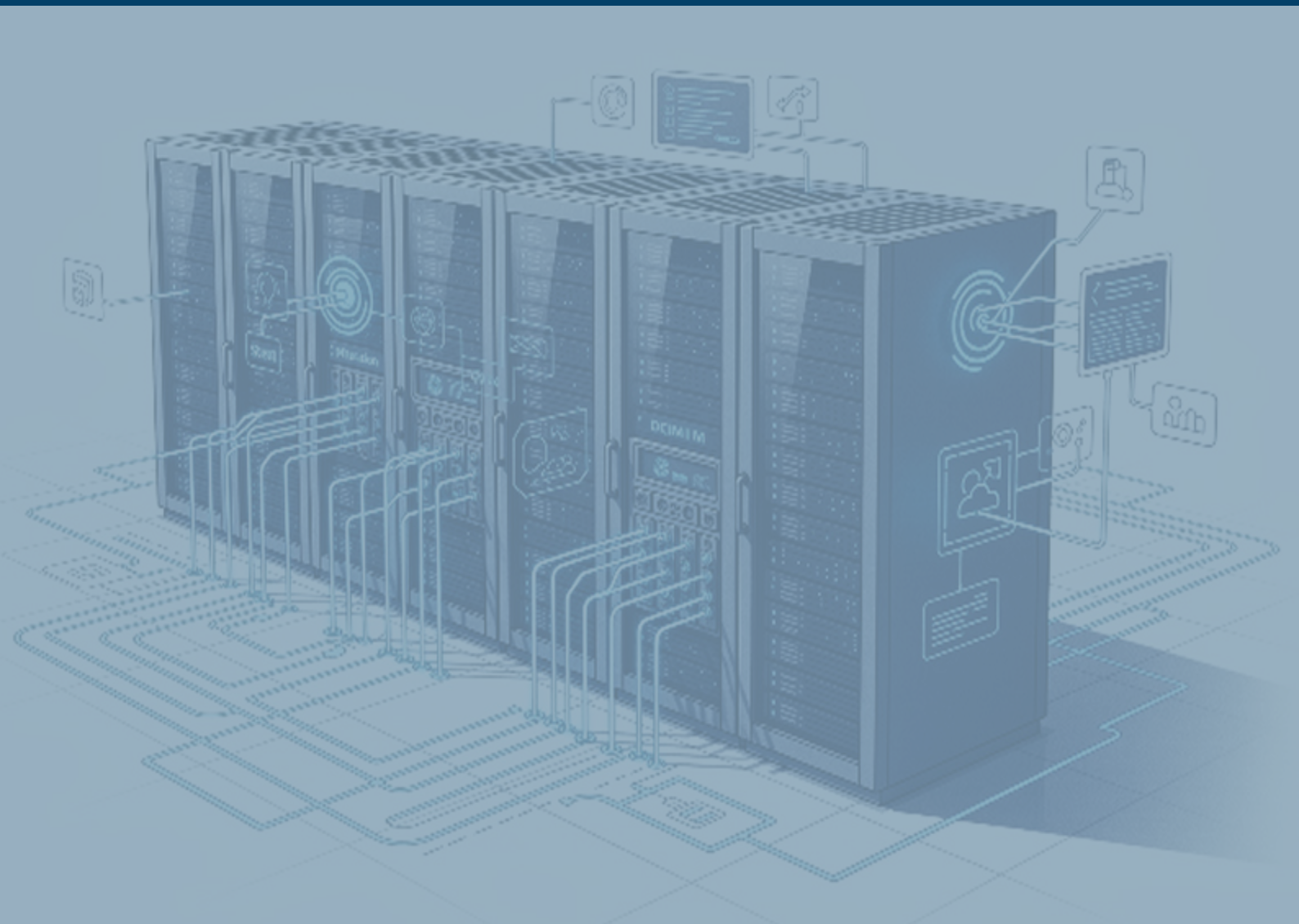
Bjumper

Riesgos en la implantación de un DCiM y estrategias para mitigarlos

Modelar, monitorizar e integrar con criterio



Introducción

Implantar un DCiM no es solo un proyecto tecnológico, es un cambio en la forma de entender y gestionar la infraestructura, y como en cualquier cambio relevante aparecen dudas, resistencias y riesgos.

Es normal, lo importante no es negar que existen, sino saber identificarlos a tiempo y abordarlos con método y sentido común.

Con una aproximación práctica y bien estructurada, esos riesgos dejan de ser una amenaza y pasan a convertirse en parte natural del proceso de madurez operativa.

Un DCiM aporta valor cuando consigue tres cosas a la vez:

1	2	3
Describe bien tu realidad (modelar)	**Mide** bien lo que ocurre (monitorizar)	**Conecta** este dato con el resto de la organización (Integraciones)

No es “instalar una herramienta”, es **convertir conocimiento disperso en operación controlada.**

Capas del proyecto

Una forma de controlar los riesgos de un proyecto de DCiM es atender a las diferentes capas o fases, que no tienen por qué ser consecutivas, sino que pueden realizarse en paralelo, pero siempre con seguimiento paralelo.

Vamos a ver las primeras, aunque esto solo es cuestión de criterio.

Modelar

Construir el “mapa” del Data Center:

Lo primero que uno se tiene que preguntar es, **¿tengo la información que requiere el modelado? ¿Dónde se encuentra? ¿Quién lo tiene? ¿Está actualizado?**

Información que se requiere:

- Estructura: site / sala / fila / rack.
- Cadena eléctrica (UPS, tableros, PDUs, racks)
- Definición clara de relaciones (qué depende de qué).
- Inventario TI (y/o infra según alcance).
- Estándares de nomenclatura y clasificación.
- Conectividad base.

¿Esto significa que tienes que tener todo antes de comenzar un proyecto DCiM? No, gran parte lo vas a ordenar y levantar en el desarrollo del DCiM, lo que significa es que tienes que ser consciente antes de comenzar de qué es lo que tienes y qué te falta.

Riesgo habitual: Modelar de forma incompleta o desactualizada convierte el DCiM en una “foto bonita” pero inútil para la operación, no incluyas datos de los que no tengas confiabilidad.

Entrega (hito de calidad): Modelo coherente vs realidad:

• Layout: Tamaños y distribución de salas, huellas de suelo de equipos de facilities, racks, distribución eléctrica (si es por elevación) y distribución de aire (baldosas perforadas/ductos elevados/Inrows/pasillos confinados).

• Trazabilidad eléctrica: Cómo llega la energía desde la entrada (proveedor/generador) hasta eslabón final (rPdus/borneras/ficha steck/equipos de AA) teniendo en cuenta redundancias.

• Trazabilidad networking: Como llega la conectividad desde la entrada (Carriers/oficinas) hasta el eslabón final (computo/storage).

• Estándar de naming IT: DC/sala/fila/rack/posición U.

• Datos mínimos completos de activos: Ubicación, Tipo, marca, modelo, IP, etiquetas.

Monitorizar

Tomar datos directos del hardware por protocolos estándar (lecturas vivas y fiables):

Información que se requiere:

- Captura fiable de consumos eléctricos, temperatura, carga, estado de equipos.
- Umbrales coherentes con la realidad operativa.
- Alarmas accionables (no ruido constante).
- Correlación entre eventos físicos y lógicos.

Riesgo habitual: Exceso de datos sin análisis equivale a una saturación operativa y pérdida de confianza en la herramienta.

Entrega (hito de calidad)

• Señales correctas (valor/unidad/escala).

• Estabilidad de comunicación: Correcta configuración de parámetros de red y equipos. Tiempo entre pooleo de datos acorde a la cantidad de dispositivos.

• Alarmas con sentido: Correcta configuración de políticas de notificación según criticidad. Evitar pantallas o casillas de mail sobrecargadas de alarmas que no son críticas (umbrales, desconexiones programadas, entrada en mantenimiento, etc).

• Datos mínimos completos de activos: Ubicación, Tipo, marca, modelo, IP, etiquetas.

Integraciones

Un DCiM aislado pierde gran parte de su potencial, el verdadero valor aparece cuando el dato fluye hacia otros sistemas. Aquí es donde el DCiM comienza a convertirse en el cerebro del Data Center.

Integración BMS con DCiM (unidirecional)

Qué aporta el BMS

datos de facilities (clima, energía/estado en ciertos elementos, alarmas del entorno, etc. según cada BMS).

Qué aporta el DCiM

Contexto operacional:

- "¿Dónde ocurre?"
- "¿Qué racks/equipos dependen?"
- "¿Qué servicio puede verse afectado?"
- "¿Qué capacidad/energía está comprometida?"

Riesgo habitual

integración "funciona" pero no es útil (eventos sin contexto o métricas mal interpretadas).

Buenas prácticas

- definir qué señales "van a operación" (las que realmente se miran).
- normalizar unidades y umbrales desde el principio.
- piloto: 1 sala / 1 conjunto de puntos ---- validar ---- escalar.

Integración CMDB con DCiM (bidirecional)

Qué aporta el CMDB (lógico)

- servicio, aplicación, owner, criticidad, estados IT
- relaciones lógicas (CI a servicio, dependencias).

Qué aporta el DCiM (físico)

- ubicación (site/sala/rack/U).
- dependencias eléctricas, puertos físicos / patching (si aplica).
- contexto físico (entorno, energía, capacidad).

Valor real

cerrar el gap entre "lo que TI cree que tiene" y "lo que físicamente existe y soporta el servicio".

Riesgos típicos

- conflicto de "source of truth" (¿quién manda en qué campo?).
- datos duplicados o que se pisan (naming, estados, owners).
- bidireccional sin reglas --- se convierte en "ping-pong de datos".

Buenas prácticas

- Definir ownership por campo (esto evita el 70% de problemas).
  Ejemplo típico:
  Ubicación física ---- DCiM manda.
  Owner / servicio / criticidad --- CMDB manda.
- Definir el "ID común" (serial, asset tag, CI ID…) para reconciliación.
- Definir eventos de sincronización (alta, baja, cambio, movimiento).
- Criterio de aceptación: "coinciden los activos clave y no se pisan campos".
- Diseñar integraciones basadas en casos de uso reales, no en "posibilidades técnicas".

Cuando se integra correctamente, el DCiM deja de ser una herramienta de infraestructura y se convierte en un sistema de soporte a la toma de decisiones.

Los bloques que más se repiten

Prerrequisitos y accesos

Antes de empezar con el DCiM, necesitamos que la base esté preparada y aquí es donde más bloqueos aparecen.

- Arquitectura y plataforma listas (servidores provisionados, versiones validadas, recursos dimensionados correctamente).
- Licencias disponibles y correctamente asignadas.
- Acceso remoto operativo (VPN, saltos, credenciales).
- Comunicaciones habilitadas entre redes implicadas (IT, OT, gestión).

Muchas veces no es un problema técnico complejo, sino pequeñas dependencias que nadie ha cerrado formalmente.

Regla:
Esta fase puede frustrar bastante. Una forma muy útil de desbloquear es identificar rápidamente a las personas adecuadas dentro de tu organización y ponerlas en contacto directo con las personas del proveedor que necesitan esa validación. En Bjumper, al menos, trabajamos así, menos correos cruzados y más conversación directa y si es presencial, mejor todavía.

Toma de datos para modelar

El modelado depende directamente de la calidad del dato que recibimos.

- Inventario y layout actualizados + sesiones de aclaración de dudas.
- Conectividad de red documentada (qué conecta con qué) + revisión conjunta para resolver incoherencias.
- Documentación eléctrica (esquemas unifilares actualizados).
- Planos CAD y cualquier detalle físico necesario para representar correctamente sala, racks y distribución.

En la práctica, casi siempre aparecen diferencias entre lo documentado y la realidad, y es normal, lo importante es detectarlas antes de cargar el modelo.

Regla:
El DCiM no arregla datos malos, los amplifica. Si el inventario está desalineado, el error no desaparece por el contrario se hace más visible.

Monitorizar: medir bien antes de medir todo

Uno de los errores más habituales es querer integrar todas las señales desde el primer día. Para que la monitorización funcione:

- Los protocolos deben ser estándar y estar bien configurados (por ejemplo, SNMP correctamente habilitado).
- Las tablas SNMP o MIBs deben estar identificadas, accesibles y documentadas.
- Las credenciales y comunidades deben estar validadas.
- Las señales críticas deben estar probadas antes de escalar.
- Decidir qué datos realmente aportan valor al DCiM. No todo lo que se puede medir se debe integrar.

Por ejemplo:

Tiene sentido integrar consumo por PDU, carga por rama y temperatura por pasillo si queremos hacer capacity planning energético.
No tiene sentido empezar integrando 200 sensores secundarios si aún no tenemos claro qué KPI queremos explotar.

Regla:
Primero “pocas señales, bien”. Luego escalamos.

Integraciones: el truco es acotar

Integrar con el BMS, con la CMDB o con cualquier otro sistema no es un hito en sí mismo. La integración técnica no es el objetivo. El verdadero hito es tener un caso de uso operativo funcionando, con criterio de aceptación claro.

Por ejemplo:

- Que un cambio en la CMDB actualice automáticamente el modelo físico validado.
- Que una alarma crítica del BMS genere un evento correlacionado y accionable en el DCiM.
- Que una ampliación de carga impacte automáticamente en un informe de capacidad.

Si no hay un caso de uso concreto detrás, la integración se convierte en un “check” de proyecto sin impacto real.

Regla:
No integrar por integrar. Integrar para resolver algo concreto.

Cómo convertir riesgos en hitos accionables

Un riesgo en un proyecto DCiM no desaparece por identificarlo en un documento. Se reduce cuando lo transformamos en algo concreto, medible y con responsable.

La diferencia entre un proyecto que avanza y uno que se bloquea suele estar aquí, en cómo pasamos de “esto puede ser un problema” a “esto tiene dueño y fecha”. Para cada hito, especialmente en datos, monitorización e integraciones lo mínimo que debe existir es:

Dueño:

Una persona concreta, no un área, no “el equipo de IT”, no “Facilities”.
Se requiere nombre y apellido debido a que, si algo no tiene dueño, en la práctica no existe y si tiene varios dueños, probablemente nadie lo priorice.

Fecha:

Fecha realista y acordada, no impuesta. Es mejor una fecha consensuada que una fecha optimista que se incumpla tres veces. Además, conviene diferenciar entre:

- Fecha de entrega técnica.
- Fecha de validación funcional.

Porque no siempre coinciden.

Dependencias:

Aquí es donde más sorpresas aparecen, porque un hito de integración puede depender de:

- Apertura de firewall.
- Entrega de credenciales.
- Activación de SNMP.
- Validación de un inventario previo.
- Disponibilidad de un entorno intermedio.

Si las dependencias no están explicitadas, el retraso parece “inexplicable”, cuando en realidad estaba anunciado desde el principio. Ponerlas por escrito ayuda a que todos entiendan qué tiene que pasar antes de dar el siguiente paso.

Crieterío de aceptación

Este es el punto más importante y el que menos se define, ya que no basta con decir “integración completada”. La pregunta es: ¿qué tiene que ocurrir exactamente para considerar que está bien?

Ejemplos:

- ¿Se reciben datos?
- ¿Se reciben con la periodicidad correcta?
- ¿Están mapeados correctamente en el modelo?
- ¿Generan alarmas coherentes?
- ¿El usuario final los puede explotar?

Si no definimos esto antes, la validación se convierte en una discusión subjetiva.

Delivered ≠ Validated

Que algo esté entregado no significa que esté validado.

- Puede estar instalado, pero no probado en carga real.
- Puede estar integrado, pero sin datos consistentes.
- Puede estar modelado, pero sin revisión por parte del responsable de sala.

La validación implica que:

- El responsable lo ha revisado.
- Se ha probado en condiciones reales.
- Cumple el criterio de aceptación acordado.

Hasta que eso no ocurre, el riesgo no está cerrado. Solo está “movido”.

Convertir riesgos en hitos accionables no es burocracia de proyecto, es la forma más práctica de evitar bloqueos largos y silenciosos. Cuando hay dueño, fecha, dependencias claras y criterio de aceptación definido, el proyecto gana tracción y el DCiM deja de ser una iniciativa ambiciosa para convertirse en una implantación controlada.

Checklist

Este checklist no es teórico, es lo mínimo que debería estar cerrado antes de dar cada bloque por “completo”. Si falta alguno de estos puntos, lo normal es que el problema aparezca más adelante.

Modelar

Estándar de naming y jerarquía

Definir cómo se nombran salas, racks, PDUs, equipos y circuitos antes de empezar a cargar información. Debe existir una jerarquía clara (site > sala > fila > rack > equipo, por ejemplo) y reglas homogéneas.
Si cada área nombra de forma distinta, el modelo acaba siendo inconsistente y difícil de mantener.

Dataset de inventario/layout completo

No solo un Excel con equipos, debe incluir:

- Ubicación física exacta.
- Consumo nominal (si aplica).
- Conectividad básica.
- Estado operativo.

Cuanto más estructurado venga el dataset, menos retrabajo habrá después.

Documentación eléctrica mínima (según alcance)

No siempre hace falta el nivel máximo de detalle, pero sí al menos:

- Unifilares actualizados.
- Identificación clara de líneas, protecciones y redundancias.
- Relación entre rack y alimentación real.

Si el alcance incluye capacidad energética, este punto deja de ser opcional.

Sesión de validación conjunta del modelo

Antes de dar el modelado por cerrado, debe haber una sesión con quien conoce realmente la sala. No basta con que el modelo “cuadre en la herramienta”. Tiene que cuadrar con la realidad operativa puesto que, una revisión conjunta ahorra muchas correcciones posteriores.

Monitorizar

Lista de fuentes HW y señales prioritarias

No se trata de integrar todo desde el inicio, debe existir una lista clara de:

- Qué dispositivos se integran primero (PDUs, UPS, climatización…).
- Qué señales son críticas (consumo, carga, temperatura, estado).
- Qué señales son secundarias y pueden esperar.

Esto evita saturar el sistema y al equipo.

Conectividad / red / permisos

Muchas integraciones fallan no por configuración, sino por permisos incompletos. Por esta razón aquí se validan:

- Accesos SNMP o protocolos definidos.
- Aperturas de firewall necesarias.
- Credenciales y permisos confirmados.
- Latencia y estabilidad de red.

Piloto validado (unidad / escala / estabilidad)

Antes de escalar, debe existir al menos un piloto funcionando correctamente:

- Datos coherentes en unidades (kW, A, °C…).
- Periodicidad correcta.
- Sin pérdidas intermitentes.

Si el piloto no es estable, escalar solo multiplica el problema.

Plan de escalado

Definir por fases:

1. Qué se integra después del piloto.
2. En qué orden.
3. Con qué criterio de validación.

Escalar sin plan suele generar ruido y sobrecarga operativa.

Integraciones

Aquí es donde más se tiende a simplificar, y donde más conviene concretar.

BMS con DCiM (unidirecional)

Se debe definir:

- Señales realmente prioritarias.
- Unidades correctas y coherentes.
- Umbrales alineados con operación real (no valores genéricos).
- Piloto previo antes de integración masiva.

Integrar 500 señales mal parametrizadas no aporta valor.
Integrar 20 bien definidas sí.

CMDB con DCiM (bidirecional)

Aquí el control debe ser todavía mayor.
Debe existir:

- Ownership por campo (quién es maestro de qué dato).
- Identificador común único (ID compartido y estable).
- Reglas claras de sincronización (unidireccional o bidireccional).
- Prueba de no-colisión antes de pasar a producción.

Si no se define quién manda sobre cada campo, la sincronización genera conflictos silenciosos que se detectan meses después.

Este checklist no busca burocracia, busca evitar retrabajo, discusiones posteriores y “esto pensábamos que estaba cerrado”. Cuando estos puntos están claros, el proyecto fluye. Cuando no lo están, el DCiM deja de ser una solución y se convierte en una fuente nueva de incidencias.

El objetivo de este documento es revisar la idea de que un proyecto DCiM es complejo en su desarrollo, si se siguen estos pasos y existe una colaboración entre el que implanta y el usuario del sistema se conseguirá una implantación fluida donde el resultado será tener una tecnologia que soporte tus procesos y facilite la operacion a las personas.

Así como dará visibilidad y control del estado del Data Center a presente y futuro. Lo que se conseguirá será:

1. Minimizar riesgos.
2. Reducir costes.
3. Aumentar y mejorar el nivel del servicio del Data Center.