

Connected Service Model

CSM 1.0 | White paper

Connect service commitments to the systems, resources and work that deliver them.

The Connected Service Model provides a shared vocabulary and relationship structure for services, service offerings, applications, deployed service instances, configuration items, assets and service requests.

It is designed for progressive adoption: an organisation can begin with clear service ownership and support commitments, then add technical relationships and operational controls where they create practical value.

Version: 1.0 - proposed reference baseline

Date: 2 October 2026

Audience: service owners, IT managers, service desk teams, configuration and asset managers, architects and implementation teams.

This document defines the proposed CSM model and recommended practices. Implementation notes describe behaviour evidenced in the supplied design documents; they are not a source-code audit or a certification of production behaviour.

Reading this white paper

CSM is a service information model. It defines what entities mean, how they relate, and how operational work can use those relationships. It does not prescribe a particular software vendor, database technology or organisational structure.

The main chapters provide model guidance. **Recommended practice** identifies a proposed CSM rule or adoption practice. **Implementation note** identifies a capability, limitation or behaviour described in the supplied implementation documentation. These labels prevent recommendations from being mistaken for already implemented features.

Topic guide

Topic	Chapter
Purpose, scope and expected outcomes	1
Modelling principles	2
Service lifecycle and entity lifecycles	3
Conceptual model	4
System model	5
Shared foundation	6
Service portfolio and commitments	7
Application design	8
Configuration and delivery	9
Relationship semantics and impact	10
Service consumption and fulfilment	11
Assets and supply	12
Operations and knowledge	13
Routing, change controls and availability	14
Governance and data quality	15
Logical entities and relationships	16
Adoption stages	17
Migration	18
Reporting and practical scenarios	19-20
Frequently asked questions	21
Implementation notes and scope boundaries	22
Glossary and supporting material	23

The diagrams are relationship views rather than database schemas or process maps. The model name is **Connected Service Model**, abbreviated **CSM**.

1. Purpose, vision and scope

CSM connects the customer-facing description of a service to the commitments, deployed systems, resources and operational work needed to support it. Its purpose is to make service information understandable and usable across service management, configuration management and asset management.

The model answers six practical questions: What service is involved? Which offering applies? Who is accountable and who handles the work? Which deployed system is affected? What resources does that system depend on? Which request, incident, problem, change or knowledge record provides the operational context?

Expected outcomes

- A consistent distinction between a service, an offering, an application, a deployment and an asset.
- More reliable assignment and support commitments through offering context.
- Better incident and change assessment through explicit technical relationships.
- Clear separation of financial/logistical asset information from operational configuration information.
- A service catalog connected to fulfilment and relevant service context.
- Progressive adoption without requiring every organisation to build a complete technical map first.

Scope

The CSM core contains Services, Service Offerings, Business Applications, Service Instances, Configuration Items and their relationships. Connected domains add reference data, service requests, assets, suppliers, contracts, knowledge and operational measurements.

CSM does not currently define a strategic planning model, business capability architecture, information-object portfolio, software development lifecycle, software bill of materials, specialised AI governance model, or full software licence-compliance model. A specialised operational technology model is also outside the baseline. Such subjects may be integrated later through deliberate extensions.

The model and its implementation

The model describes information and semantics. An implementation supplies forms, APIs, authorization, reporting and automation. A field existing in a database does not establish a complete operational capability. A usable service map requires maintained records and meaningful relationships, not merely enabled screens.

Recommended practice: adopt the smallest coherent set of entities that answers a real operational question. Expand the scope when an owner, data source and consuming process are ready to maintain and use it.

2. Modelling principles

Separate meanings before connecting records

A Service describes a supported outcome or capability. An Offering describes a specific scope and support commitment. A Business Application describes an application independent of its deployments. A Service Instance describes a concrete operational deployment. A CI describes an operational resource. An Asset describes administrative, financial or logistical custody.

These distinctions prevent a requestable action, a deployment, a support package and a purchased item from becoming interchangeable records.

Make relationships explicit

Store relationship endpoints, type and operational meaning. A generic link is insufficient to decide whether a failure causes unavailability, degradation or only potential impact. Relationship direction and impact direction need deliberate definitions.

Start with commitments and accountability

Services and offerings need accountable owners. Offerings provide the normal support context: assigned team, scope, environment, schedules and priority-based response/resolution targets. Technical detail should strengthen that context rather than substitute for it.

Keep adoption progressive

Optional links support staged population. They should not become an excuse to leave critical data incomplete indefinitely. Define completeness criteria appropriate to the organisation's current adoption stage.

Preserve distinct sources of truth

Assets own purchase, contract, assignment and stock information. CIs own operational configuration. A reconciliation process identifies disagreement; it does not automatically make either record authoritative for every field.

Distinguish targets, policies and measurements

A reusable target is not an individual SLA measurement. An availability policy is not an outage. A planned change is not automatically downtime. Measurement scope, time basis and counting rules need explicit definitions.

Govern change and extension

Use stable identifiers, defined classifications and a versioned relationship vocabulary. Configuration should adapt vocabulary and workflows without silently changing the meaning of a core entity.

Recommended practice: publish an entity dictionary and relationship matrix, assign data stewardship, and document each extension's purpose, owner and consumers before introducing it.

3. Lifecycle and adoption of service information

CSM describes a service information lifecycle without imposing a single status enumeration on every entity. The lifecycle below is recommended guidance; it is not a claim that an end-to-end workflow engine already enforces these stages.

Stage	Information and decisions	Completion evidence
Define	Name the service, scope, owner and consumers	Agreed service purpose and accountability
Package	Define offerings, support teams, schedules and commitments	A usable support package
Connect	Register applications, deployments and relevant CIs	Reviewed links and dependency meaning
Publish	Expose appropriate catalog actions and instructions	Requestable actions with fulfilment ownership
Operate	Use records in incidents, requests, changes and outages	Traceable work and measurements
Improve or retire	Review quality, usefulness and remaining obligations	Updated scope or controlled retirement

These stages can repeat. A service may operate while a non-critical dependency map is still being expanded. A new offering can be added without replacing the service or its applications.

Entity lifecycles remain distinct

Services and offerings describe portfolio/support lifecycle. CIs describe operational and lifecycle status. Assets describe custody and administrative lifecycle. Knowledge describes content approval, publication and retirement. Tickets describe work progression.

Implementation note: the documented asset lifecycle defaults are Planned, Ordered, Received, Active, Retired, Disposed and Cancelled. A separate operational/availability list includes In Stock, Assigned, In Use, In Repair and other states. Both lists are tenant-configurable. CI lifecycle and operational status are separate fields, not automatically synchronized with asset status.

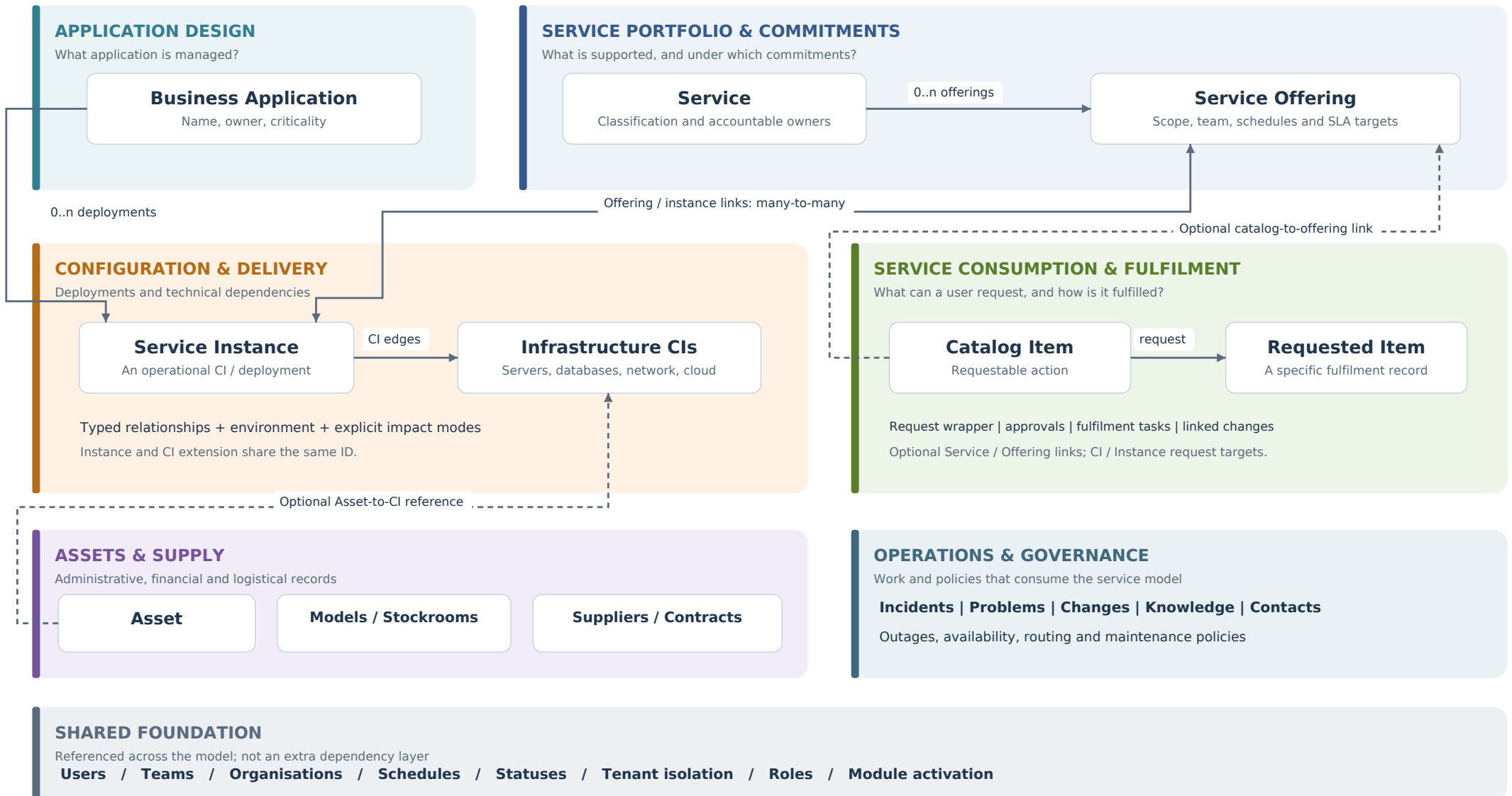
Retirement guidance

Before retiring an offering, review catalog links, open work, default-offering references and instance relationships. Before retiring a CI, review dependencies and asset references. Preserve historical evidence according to retention requirements. Closed tickets should not lose their meaning simply because a current support package changes.

Recommended practice: define permitted transitions, mandatory evidence and historical-reference rules for each entity family. Labels alone are not reliable workflow logic when users can rename statuses.

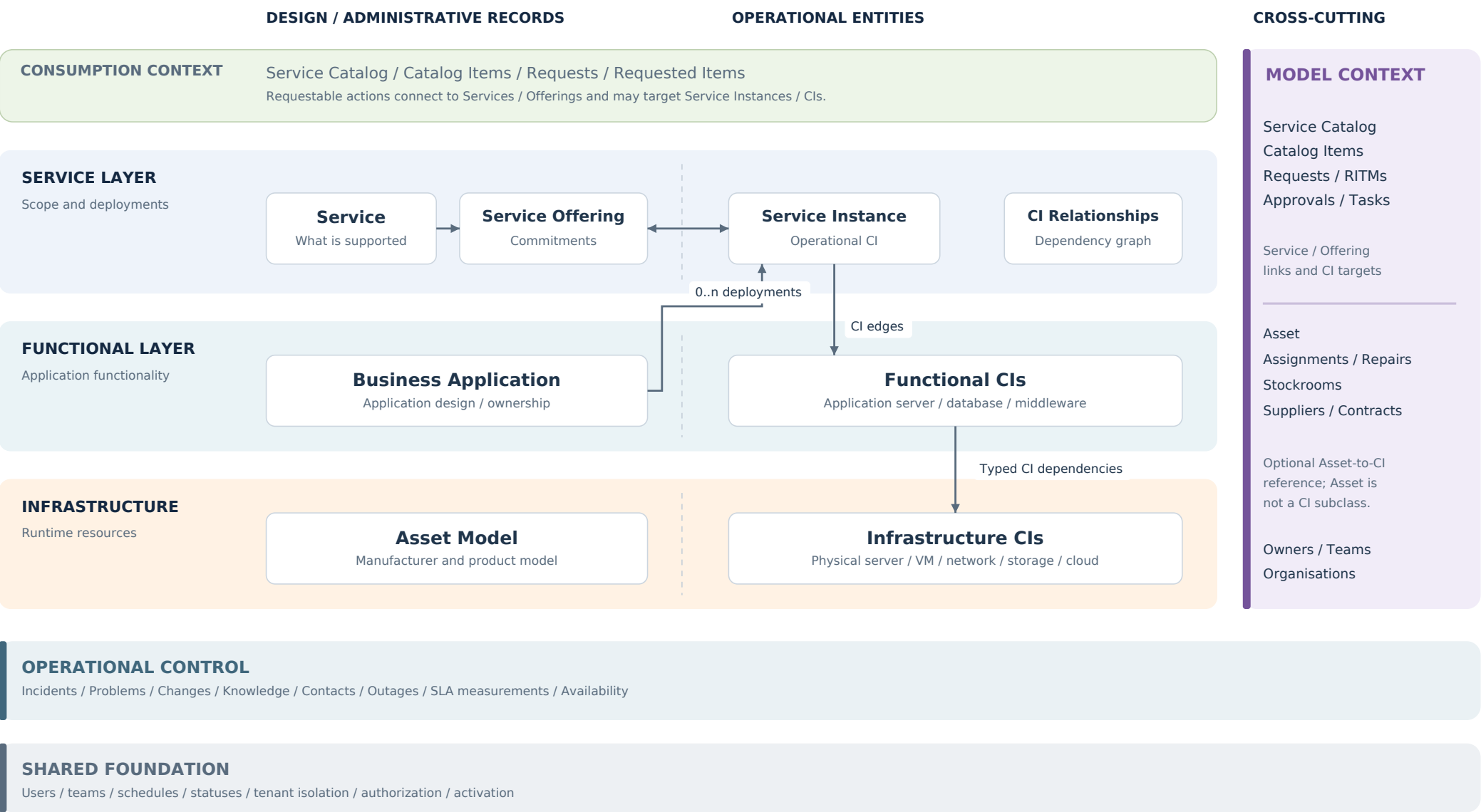
4. Conceptual model

A domain view of services, deployments, requests and assets - connected by a shared operating foundation.



5. System model

A layered view of design / administrative records and operational entities - showing the separation of design, commitment and runtime concerns.



6. Shared foundation

Foundation data gives the model a common operational context. It includes users, teams, organisations, suppliers, schedules, statuses and priorities. Asset categories, models and stockrooms add administrative reference data. These objects are referenced by other domains; they are not all nodes in a technical dependency graph.

People, teams and organisations

Users identify requesters, assignees, owners and approvers. Teams represent groups responsible for handling work. Organisations group teams for accountability and organisational measurements. A supplier identifies an external provider; it is not necessarily the same entity as an organisation containing support teams.

Recommended practice: define a controlled mapping where supplier responsibility and organisational assignment must be analysed together. Do not assume equivalence from matching names.

Schedules, statuses and priorities

Schedules establish support hours, availability commitments and maintenance windows. A schedule needs a timezone and a clear recurring pattern. Statuses communicate entity progression; priorities support differentiated operational targets.

Implementation note: recurring weekly schedules, including overnight windows, are documented. The business-time SLA calculation does not include a holiday calendar. A support-hours schedule and an availability commitment schedule may serve different purposes even if they happen to contain the same hours.

Location and product references

Stockrooms are named storage locations for assets. Free-text asset location is not equivalent to a general location hierarchy. Asset Models identify manufacturer/model combinations and categories; they are not versioned software-development components.

Tenant and access context

Implementation tenant boundaries, user roles and module activation govern who can see and change records. Access rights are distinct from service ownership, technical dependency and support responsibility.

Recommended practice: keep canonical identifiers stable, normalize duplicates before importing, and assign a steward for every shared register. Prefer a controlled reference over free text when repeated values need reliable reporting, but do not create new entities without a maintenance need.

7. Service portfolio and commitments

Service

A Service is a named outcome or capability that an organisation provides or supports. It provides a stable context for ownership, scope and operational reporting. Services may be business-facing or technology-facing; the CSM baseline uses a generic entity with classification rather than requiring separate entity classes.

A good name describes the supported purpose: Employee Collaboration, Customer Ordering or Database Hosting. An incident title, a request action or a server hostname should not normally become a Service name.

Service Offering

A Service Offering is a specific support package or variation of a Service. It can distinguish coverage, environment, criticality or support arrangements. Examples include Business Hours Support and Production Database Hosting.

Information	Service	Service Offering
Purpose and broad scope	Primary location	More specific scope
Accountability	Owner / secondary owner	Owner / secondary owner
Normal work assignment	Through default offering	Assigned team
Response / resolution targets	Context only	Priority-based targets
Support and maintenance schedules	Broader policy context	Offering schedules
Environment and criticality	Broad classification	Offering-specific values

Defaults, hierarchy and readiness

A Service can have zero or more Offerings. A default Offering provides normal incident support context. Creating a Service without an Offering may be useful during setup, but it should be visibly incomplete until its required support information exists.

Recommended practice: designate an active, complete Offering before promoting a Service for operational selection. Verify that a default Offering belongs to its parent Service. If service hierarchies are used, define their meaning and prevent cycles; hierarchy alone should not imply commitment or impact inheritance.

Implementation note: the supplied schema includes an optional parent Service and a default Offering reference. It permits initial Service creation without an Offering. Full hierarchy validation is not established by the documents.

8. Application design

A Business Application is an application-level record independent of individual environments or deployments. It identifies what application is managed and who is accountable for it. Examples include Employee Portal, Finance Application or Customer Relationship Management.

The CSM baseline application record is deliberately lightweight: name, description, owner and criticality. A Business Application can have several Service Instances, such as Employee Portal PROD, Employee Portal TEST or a regional production deployment.

Application versus deployment

Question	Business Application	Service Instance
What does the record identify?	An application managed as a whole	A concrete deployed system
What is its perspective?	Ownership and application context	Operation and technical dependencies
Is it the incident's operational CI?	Not in the baseline model	Yes, when selected as the affected deployment
Can it span environments?	Yes	Normally identifies a specific deployment context
Does it share the CI identity?	No in the documented implementation	Yes

Do not duplicate an application record merely because it has a development and production instance. Create separate applications when there is genuinely separate application identity and accountability, not simply another hostname or environment.

Links to the service model

The application-to-instance relationship identifies deployment context. The offering-to-instance relationship identifies support or delivery context. A direct Business Application-to-Service relationship is not part of the documented baseline; trace the connection through Service Instances and their Offering links.

Recommended practice: use a stable application naming convention and an accountable owner. Define when regional variants remain instances of one application and when separate application records are justified.

Business capabilities, information objects, architecture investment plans, source-code components and development releases are outside the current Application Design domain. Their absence does not prevent an application register from providing useful operational context.

9. Configuration and delivery

Configuration Item

A Configuration Item is an operational resource whose identity, status or relationships matter to service support and change assessment. The documented types include Service Instance, Application Server, Database, Virtual Machine, Physical Server, Network Device, Network Service Instance, Storage, Middleware, Cloud Service, Container Cluster and Other.

Service Instance

A Service Instance is a concrete operational deployment or connected system. It represents the system on which incident, change and availability work operates. It can carry an application reference, region and deployment identifier. Its CI and extension record share one identity.

A Service Instance is not an Offering. One instance can participate in several offering relationships, and one offering can relate to several instances. A production deployment should not be copied solely to represent two different support packages.

Environment and status

The documented broad environment classes are PROD, NON_PROD, SHARED and UNKNOWN. A more specific environment type can identify DR, PREPROD, UAT, QA, TEST, DEV, TRAINING or SANDBOX. SHARED is useful for resources that span contexts; UNKNOWN is a data-quality signal rather than a meaningful environment design.

Lifecycle status and operational status are separate. Criticality should be used consistently and should not be inferred solely from the name of an environment.

Building a useful delivery map

Start with a critical deployment and the components that materially affect its operation. Record application servers, database dependencies, network paths and external resources at the level needed for support decisions. Avoid mapping every low-level resource before a consuming process is ready to use the result.

Recommended practice: validate the map with the responsible technical team, record the relationship's operational meaning, and review it after significant change.

Implementation note: mapping-method and calculation-related fields do not prove an automatic discovery capability. The supplied asset description explicitly excludes automatic endpoint discovery. A general discovery connector, dynamic-query grouping system or full set of service-instance subclasses should not be assumed from type names alone.

10. Relationship semantics and impact

CSM distinguishes reference relationships from technical dependency edges. Ownership and assignment explain responsibility. Offering-to-instance links explain a support/delivery association. CI edges describe technical structure and operational impact.

Offering-to-instance associations

The documented association can carry one of three roles: `SUPPORTED_BY_OFFERING`, `DELIVERS_OFFERING` or `MANAGED_THROUGH_OFFERING`. It also carries active and validity information. The stored pair/role combination is unique, so multiple roles for one pair are possible.

Recommended interpretation: define an instance as supported under an Offering, contributing to delivery of an Offering, or governed through an Offering, according to the chosen role. Publish exact direction and allowed combinations before treating these interpretations as an enforced contract. The implementation notes do not establish different resolver behaviour for every role.

CI relationship types

The documented vocabulary includes `RUNS_ON`, `HOSTED_ON`, `DEPENDS_ON`, `CONNECTS_TO`, `USES`, `CONTAINS`, `MEMBER_OF`, `LOAD_BALANCED_BY`, `DATABASE_FOR`, `SUPPORTS` and `DELIVERED_BY`. Define allowed endpoints and direction for each type. A relationship's label alone does not determine whether failure should propagate.

Impact mode	Intended modelling meaning
<code>HARD_DEPENDENCY</code>	Loss of a component can cause unavailability
<code>SOFT_DEPENDENCY</code>	Loss can indicate potential impact
<code>DEGRADATION_ONLY</code>	Loss can degrade rather than fully stop the parent
<code>REDUNDANT_MEMBER</code>	A peer group can tolerate some failures
<code>QUORUM_MEMBER</code>	Operation depends on a membership threshold
<code>NO_AUTOMATIC_IMPACT</code>	Do not infer impact automatically from this edge

Implementation note: graph traversal is documented as cycle-safe with a ten-hop cap. Redundant/quorum incident propagation uses an all-siblings-down simplification; configurable partial-quorum thresholds are not documented. Availability propagation excludes these descendants rather than calculating overlapping peer failures.

Recommended practice: test representative failure scenarios against the map. Distinguish computed impact from manually confirmed impact, and preserve the relationship rules that justify the inference.

11. Service consumption and fulfilment

The Service Catalog is the browsable collection of requestable actions. It gives consumers an entry point into the service model. It is distinct from the Service Portfolio: a portfolio describes what is managed and supported; a catalog presents what can be requested.

Catalog Item

A Catalog Item defines a requestable action: request access, provision equipment, change permissions or obtain information. It may link to a Service and Offering, define fulfilment responsibility, require approvals and specify a target type.

Several Catalog Items can share an Offering. One collaboration Offering could support request-access, change-permissions and remove-access items. The Offering defines scope and commitments; each item defines an action.

Request, Requested Item and Task

A Request groups a user's submission. A Requested Item records a particular requested action or deliverable. Fulfilment Tasks split the work into assigned activities. When a target is required, the request identifies the operational deployment or resource involved.

Object	Example
Service	Employee Collaboration
Offering	Collaboration - Business Hours
Catalog Item	Request Employee Portal access
Request	New starter provisioning for Sarah
Requested Item	Give Sarah Employee Portal access
Target	Employee Portal PROD
Task	Validate approval and provision access

Approvals and change controls

The documented item configuration supports separate manager and staff approval tracks. A change-link requirement can depend on configuration and production environment. The linked Change retains its own workflow.

Implementation note: catalog references are optional. CI and Service Instance targets are functional; some other target types are placeholders. A required Change link is satisfied by a linked record, not a check that the Change has been completed or approved. Do not present that link gate as proof of change authorization.

Recommended practice: link recurring catalog actions to their Service/Offering where possible, define required target semantics, and review completion evidence. Employee equipment profiles can generate category-specific Requested Items that use the standard asset assignment/return workflow.

12. Assets and supply

An Asset records an item's administrative, financial and logistical life. It can describe purchase, supplier, contract, ownership, assignment, stockroom, repair and disposal evidence. A CI records operational configuration. These records can refer to the same real-world resource without having the same lifecycle or source of truth.

Asset Model, Category and Stockroom

An Asset Model identifies a manufacturer/model combination. A Category groups items such as laptops, monitors or phones. A Stockroom represents physical storage and can define minimum and desired category stock levels.

Assignment histories record custody. Transfers record movement, optionally using dispatch/receipt confirmation. Inventory Campaigns compare expected stock with physically confirmed items. Repair records describe repair, return-material authorization or warranty work separately from asset lifecycle.

Supplier and Contract

A Supplier identifies the provider of goods or services. A Contract records an agreement, validity dates and cost information. The documented Asset references one optional Contract; one Contract can cover multiple Assets. Supplier is shared with other service-model records.

Asset-to-CI linking and reconciliation

An Asset can optionally reference one CI. The documented link does not enforce one Asset per CI: multiple Assets can reference the same CI, and reconciliation flags that condition. Not every Asset requires a CI, and not every CI represents a purchased physical Asset.

Recommended practice: define field-level authority before importing external data. Preserve financial and contractual information when technical observations arrive. Review missing references, inconsistent lifecycle states and duplicate links rather than silently overwriting either record.

Implementation note: assigning an Asset clears its stockroom reference and records the previous stockroom in the assignment history. Return restores that stockroom. Some lifecycle-transition and reconciliation rules match status names; those heuristics need care when tenants rename statuses.

Software entitlements, licence consumption, depreciation, accounting, endpoint management and physical disposal execution are not part of the baseline. CSM can retain evidence and references without replacing specialised systems that perform those activities.

13. Operations and knowledge

Operational records consume the service model. They should point to relevant services, offerings, CIs and assets rather than create competing definitions of those objects.

Incident, Problem and Change

An Incident records an interruption or support issue. A Problem investigates an underlying cause or recurring pattern. A Change records a controlled modification, its targets, assessment, authorization and work. These records can have primary or affected CIs and affected Assets.

The model supports both technical and service context. A primary database CI can identify the component under investigation while computed links identify potentially affected Service Instances and Offerings.

Implementation note: documented Problem suggestions group incidents by CI or error code and require human promotion or dismissal. This is not equivalent to comprehensive time-windowed trend analysis or automatic root-cause determination.

Knowledge and instructions

Knowledge Articles provide reusable guidance, known errors, workarounds and operational runbooks. Knowledge Bases define content organization and visibility. A Service link provides context and publishing accountability. Applicability rows can provide CI, environment and version context for relevance.

Implementation note: applicability influences known-error scoring, not content authorization. Public/internal base visibility and publishing eligibility remain separate. Some content lifecycle actions are manual, including scheduled publication in the documented baseline.

Contact and Ticket

A Contact represents a conversation or intake interaction. It is not automatically a Ticket. A Contact can link to several Tickets, and a Ticket can be linked from several Contacts. Closing the conversation does not automatically resolve the operational work.

This separation allows a service desk to handle questions, duplicate reports and requests for information without creating a ticket for every exchange.

Recommended practice: retain the reason for significant references, distinguish public communication from internal notes, and define who confirms operational impact. A record being linked to a service does not automatically establish outage duration, commitment breach or causal responsibility.

14. Policies, routing and measurements

Different policy families combine differently. CSM should not describe them as one universal inheritance rule.

Concern	Documented behaviour
Incident routing	Explicit enabled primary-CI override, then one-hop parent-instance override, then Offering team
Maintenance controls	Every applicable required window must be satisfied
Blackouts	Applicable restrictions accumulate; emergency exceptions require the documented conditions
Availability policy	Offering context takes precedence over Service and tenant defaults in documented scope resolution
Offering SLA	Priority target copied into individual incident measurements
Organisation SLA	Separate measurements pause outside that organisation's teams and resume on return

Commitments and SLA measurements

Response and resolution targets are configuration. Individual incident measurements record start, pause, completion and breach information. The documented implementation freezes targets and schedule context at measurement start; later target edits do not retroactively rewrite that measurement.

Outages and availability

An Outage records actual unavailability or degradation. It may relate to a CI, Instance or Offering. An Incident alone does not automatically create downtime. Availability uses committed time, recorded outage duration, applicable policy and degradation weighting.

Conceptually, availability equals $100 \times (\text{committed time} - \text{counted downtime}) / \text{committed time}$. Define behaviour for zero committed time, open outages and incomplete evidence rather than inventing a valid percentage.

Implementation note: optional propagation incorporates descendant CI outages for a specific CI/Instance target. It does not establish full automatic aggregation of all linked Instances into Offering availability. Overlapping outages are summed independently, potentially counting the same interval more than once. Redundant/quorum descendants are excluded from availability propagation.

Recommended practice: state measurement scope and evidence limitations alongside the result. Define treatment of planned work, degradation, overlapping events and business-time calendars before using availability as a contractual report. Do not infer a full outage solely from the failure of a redundant member.

15. Governance and data quality

Governance makes the model dependable. It includes accountable ownership, stewardship, access rules, relationship review and controlled extension. A complete-looking diagram without maintained evidence can produce confident but incorrect operational decisions.

Responsibilities

Responsibility	Typical accountable function
Service purpose and portfolio scope	Service Owner
Offering commitments and support arrangements	Offering Owner
Application register and application context	Application Owner
Deployment and technical relationships	Configuration steward / technical owner
Asset custody, inventory and disposal evidence	Asset Manager
Catalog action and fulfilment design	Catalog / Request process owner
Measurement definitions and report acceptance	Service-level / process owner
Shared data and naming quality	Data steward

These are recommended responsibilities, not a proposed replacement for an implementation's authorization roles. One person may hold several responsibilities, but a broad administrator permission should not remove the need for accountable ownership.

Minimum quality dimensions

Assess completeness, uniqueness, validity, referential integrity, freshness and semantic correctness. An Offering can have a team but still have the wrong support scope. A CI edge can point to valid records but express the wrong dependency direction.

Recommended practice: establish stage-specific quality checks. A service-based operation needs complete support information. Dependency automation additionally needs validated endpoints, impact meaning, environment and redundancy interpretation.

Review and change discipline

Review critical maps after significant Change, reconcile Asset/CI disagreements, and retain decisions on exceptions. Define review ownership and cadence based on risk rather than treating every register as equally critical.

Implementation note: module activation, user roles, ownership checks and tenant isolation are separate controls. The supplied system uses selected database foreign keys and scalar references; application validation and reconciliation therefore remain important. A common, cross-domain attestation programme is recommended guidance, not an evidenced implemented feature.

16. Logical entities and relationships

The table below specifies the principal cardinalities evidenced by the design documents. It is a logical contract to validate against actual schema and services before using it as a migration or integration specification.

Relationship	Cardinality and meaning
Service - Offering	One Service has zero or more Offerings; each Offering has one parent Service
Service - Default Offering	Optional reference; recommended to select an Offering belonging to that Service
Application - Instance	One Application has zero or more Instances; an Instance has zero or one Application
Instance - CI	One-to-one extension with a shared identity
Offering - Instance	Many-to-many association, qualified by role
CI - CI	Directed edges with type, impact mode and optional context
Offering - Team	One required normal assignment Team per Offering
Team - Organisation	Optional Organisation per Team; one Organisation can group many Teams
Asset - CI	Zero or one CI reference per Asset; reverse uniqueness is not enforced
Asset - Model / Category / Stockroom	Administrative references; assignment affects stockroom association
Asset - Contract	Optional Contract per Asset; one Contract can cover several Assets
Catalog Item - Service / Offering	Optional service-context references
Request - Requested Item	A Request groups its Requested Items
Requested Item - Change	Linking table supports associated Change records
Ticket - affected CI / Asset	Separate many-to-many links
Contact - Ticket	Many-to-many association

Identity and integrity guidance

Use stable identifiers for durable links, even when names change. Validate reference existence and tenant scope. Validate association roles, default-offering parentage and active/valid relationships. Retain audit evidence for changes to important semantics.

Implementation note: delete behaviours differ. Deleting a Business Application is documented to clear Instance application references. Asset-to-CI scalar links may become orphaned and are surfaced by reconciliation. Service/Offering deletion has restrictions where defaults or ticket references remain. Do not assume one uniform cascading rule across the model.

17. Progressive adoption

CSM adoption is based on useful outcomes rather than an obligation to populate every domain. The stages below are proposed guidance, not feature flags or automatic maturity scoring.

Stage	Start with	Exit evidence
Establish	Foundation, Services, Offerings, teams and ownership	Support can select a coherent Service and Offering and route work reliably
Connect	Applications, critical Instances, CIs and reviewed relationships	A technical issue can be traced to relevant deployments and support context
Govern	Catalog actions, approvals, change controls, Assets and reconciliation	Fulfilment and lifecycle decisions use consistent references and evidence
Measure and improve	SLA/availability definitions, quality checks and operational reporting	Reports are reviewed against actual cases and trigger improvement work

Establish

Choose a small set of services that people recognize. Define owners, default Offerings, teams and priority targets. Populate common schedules and terminology. Keep the selection experience simple.

Connect

Add deployments for important applications and the dependencies needed for incident/change assessment. Resolve environment ambiguity and validate impact meaning before enabling automatic traversal.

Govern

Link recurring catalog actions to service context. Define approvals and fulfilment ownership. Connect relevant assets while maintaining separate lifecycle responsibilities. Review maintenance, blackout and retirement controls.

Measure and improve

Check measurement scope and data quality. Sample completed work and known outage scenarios. Review false impact inferences and unlinked critical resources. Expand only where information can be maintained.

Implementation note: the documented platform exposes two operating views, a simpler service-based view and a configuration-connected view, over one schema. Whole-module activation and roles remain separate. These settings do not certify an organisation's adoption stage and should not be presented as maturity scores.

18. Migration into CSM

Migration is a controlled change to entity meaning, relationships and consuming processes. Renaming screens alone is not a data migration; moving records alone does not update the reports, rules or integrations that depend on them.

1. Inventory and classify

Collect existing service lists, application registers, CI inventories, asset records, catalog actions and integration references. Classify each record according to its actual meaning. Identify deployments stored as applications, support packages stored as services and requests stored as offerings.

2. Map fields and responsibilities

Define destination entities, retained identifiers, naming rules, source-of-truth fields and owners. Record unresolved classification and duplication decisions. Preserve attributes that have an identified business purpose; retire unused fields through a reviewed decision.

3. Map consuming dependencies

Inventory forms, assignment rules, SLA configuration, reports, scripts, API contracts and integrations that reference old entities or labels. Maintain stable IDs where possible. Document compatibility mappings for renamed enum values and persisted settings.

4. Rehearse and validate

Back up source data and verify recovery. Run a representative migration in a controlled environment. Check record counts, orphaned references, duplicate identities, default Offerings, graph direction and tenant boundaries. Test practical incident, request, change and asset scenarios rather than relying on record counts alone.

5. Transition and review

Use a phased cutover with rollback criteria, owner sign-off and reconciliation. Confirm that reports and integrations resolve their references correctly. Retain historical context and communicate changed terminology to the people selecting service information.

Recommended practice: do not silently merge records or overwrite financial data using technical observations. Route ambiguity into a review queue. Recheck the model after the first operational cases reveal assumptions that were not visible in the source register.

Implementation note: the supplied documents describe some contradictory or stale plan statements. Resolve them against source code and schema before migrating. Existing persisted mode names can remain internal while customer-facing terminology is updated separately.

19. Reporting and operational value

CSM defines the information structure needed for useful reports. It does not guarantee a particular dashboard or that every logical query is exposed by a current reporting engine.

Question	Information needed	Interpretation check
Which Services lack accountable ownership?	Service owners and lifecycle	Ownership is not inferred from assignment
Which Offerings have usable support commitments?	Offering team, schedules and targets	Distinguish absent targets from missed targets
Which critical deployments lack reviewed dependencies?	Instance criticality and CI edges	A map existing is not proof of review
Which incidents breach response/resolution targets?	Incident measurements and priorities	Use frozen measurement configuration
Where does work wait across organisations?	Team/organisation history and measurements	Paused organisational time differs from total elapsed time
Which requestable actions lack service context?	Catalog Service/Offering references	Some standalone asset actions may be intentional
Which assets need reconciliation?	Asset/CI references and issue records	Acknowledgement does not correct the underlying condition
What availability was actually delivered?	Committed schedules and outage evidence	Check overlaps, propagation and redundancy limitations

Quality and interpretation

Present missing data as missing rather than as zero impact, zero breach or full availability. A period without recorded outages can reflect incomplete evidence. A CI with no dependency edges can be unmapped rather than independent.

Distinguish actual operational measurements from model coverage indicators. Coverage describes how much of the service estate has useful information; it is not itself proof of service performance.

Implementation note: the documented analytics engine uses an allowlist of entities and fields. Available dashboards and cross-entity joins must be checked against that registry. The reporting questions above are recommended uses of CSM, not a claim that all are currently available as built-in reports.

Recommended practice: agree report definitions with service/process owners, sample the underlying records and make data-quality limitations visible. Use the results to decide where modelling effort improves actual work.

20. Worked scenarios

A. Employee Portal incident

Employee Collaboration is the Service. Business Hours Support is its Offering. Employee Portal is the Business Application, with PROD and TEST Instances. The production map includes an application server, database and load balancer.

An incident identifies the database CI. When configured, dependency traversal identifies affected Instances and linked Offerings. Normal routing uses the Offering team unless an applicable explicit CI/Instance override exists. A service owner can review impact without treating the application register and the production deployment as the same record.

If actual user-facing unavailability occurs, record an Outage separately. Availability needs that evidence and the relevant counting policy; incident creation alone is insufficient.

B. New starter equipment

A catalog action links to an employee equipment profile containing laptop and monitor categories. Submission generates category-specific Requested Items. Fulfilment selects actual Assets and uses the ordinary assignment workflow.

An Asset's stockroom reference is cleared on assignment while its prior stockroom is retained in assignment history. The Asset can have an optional CI link, but financial information and custody remain asset concerns. Offboarding can drive the corresponding return workflow.

C. Production maintenance

A Change targets a production deployment and relevant CIs. The planned window must satisfy each applicable required maintenance window. Overlapping blackout restrictions also apply according to policy. A linked request does not replace the Change's own assessment and authorization.

An incident during maintenance does not automatically prove a planned outage. Record actual downtime and apply the availability policy's treatment of planned work.

D. Shared technical deployment

One deployment can participate in several Offering associations. Create one Service Instance identity and explicit links instead of copying the deployment per support package. Define which link role expresses delivery, support or management responsibility. Review environment compatibility and the resolver's treatment of association roles before relying on automatic impact results.

These scenarios are illustrative model examples, not descriptions of current tenant data.

21. Frequently asked questions

Is CSM a product?

No. It is a proposed information model and guidance baseline. A software platform can implement it, expose selected parts of it, and add documented extensions.

Must every organisation populate every domain?

No. Begin with a coherent service/support context. Add application, configuration, asset and request links as operational needs and data stewardship justify them.

Is a Catalog Item a Service Offering?

No. A Catalog Item defines an action that can be requested. An Offering defines service scope and support commitments. Several actions can refer to one Offering.

Is a Service Instance a Business Application?

No. A Business Application identifies the application managed as a whole. A Service Instance identifies a concrete operational deployment. One application can have several deployments.

Must every Asset have a CI?

No. CI links are optional and should serve an operational purpose. Assets and CIs have distinct sources of truth and lifecycles.

Does a technical failure automatically mean a Service outage?

No. Impact depends on relationship semantics, redundancy and actual effect. Potential or computed impact is not equivalent to confirmed downtime.

Does enabling configuration features make the model complete?

No. Completeness requires maintained records, meaningful links, owners and consuming processes. Feature settings are not maturity certificates.

Can CSM be extended?

Yes. Define the new entity or relation, its semantics, owners, integration behaviour and compatibility. Version the extension and preserve existing meanings.

Can the lifecycle labels be customized?

Yes, within a clearly defined implementation contract. Stable behavioural semantics should not depend solely on editable labels. Some documented rules currently use label matching and need validation when labels change.

22. Implementation notes and scope boundaries

This chapter separates the proposed model from the documented implementation. It does not certify source-code correctness. Tables, policies and UI routes must be verified before being used as a binding integration contract.

Core entity mapping

CSM entity	Documented storage
Service / Offering	service / service_offering
Application / Instance	business_application / service_instance
CI / CI Relationship	configuration_item / ci_relationship
Offering association	service_offering_instance
Catalog / fulfilment	catalog_item / request / requested_item_details / request_task
Operational work	ticket plus type-specific extension and linking tables
Assets / supply	asset / asset_model / stockroom / supplier / contract
Knowledge	knowledge_base / knowledge_article / knowledge_applicability
Outage / measurement	outage / incident_sla / incident_org_sla

Important limitations

- Routing's instance override lookup is one relationship hop; dependency-impact traversal can span multiple hops. They are not the same algorithm.
- Graph traversal has a ten-hop cap. Redundant/quorum incident impact is simplified to all peers down.
- Availability propagation skips redundant/quorum descendants and can double-count overlapping outage intervals. Offering-wide aggregation is not established by the documents.
- The availability calculation-level field is stored but not consumed by the described calculation; callers select the target.
- Some catalog target types and policy modes are placeholders or advisory. Link existence is not proof of Change authorization.
- Knowledge version-range relevance uses a text-match simplification. Scheduled publication is described as manual.
- Scalar cross-domain references can become orphaned. Some Asset lifecycle and reconciliation rules use editable status-name heuristics.

Outside the baseline

Strategic objectives, value streams, business capability and information-object registers, development components, software bills of materials, specialised AI/OT entities, dynamic CI grouping, subscription entitlement, automated discovery and full licence compliance are excluded from the current model definition. They should not be implied by a generic extension field.

23. Glossary and supporting material

Term	Meaning in CSM
CSM	Connected Service Model
Service	A named outcome or capability provided or supported
Service Offering	A specific scope and support package for a Service
Business Application	An application managed independently of individual deployments
Service Instance	A concrete operational deployment; a kind of CI
Configuration Item (CI)	An operational resource managed for support/change purposes
CI Relationship	A typed directed edge between CIs
Asset	An administrative, financial or logistical item record
Service Catalog	The browsable collection of requestable actions
Catalog Item	The definition of one requestable action
Requested Item (RITM)	A specific requested action/deliverable within a Request
Contact	A conversation or intake interaction, distinct from a Ticket
SLA target	Reusable service-level response/resolution configuration
SLA measurement	An individual timed operational record
Outage	Explicit evidence of unavailability or degradation
Foundation	Shared reference data used across the model

Evidence basis

The implementation descriptions draw on the supplied LLD (2).md, particularly sections 2.2-2.4, 2.8-2.16 and 7; SAD (2).md sections 1-5; HLD (2).md sections 2-3; ROLES.md sections 3.7 and 3.10-3.15; BACKLOG.md sections P1, P3, P4 and P8; and the asset management plan and functional/technical description, particularly their modelling, reconciliation and workflow sections.

Where introductory architecture claims or old plan checkboxes conflict with later detailed definitions, this paper uses the specific behaviour and records the limitation. The source schema, migrations and application code were not supplied for direct verification.

Version and governance

Version 1.0 establishes proposed CSM terminology, entity boundaries, domain views and adoption guidance. Before formal adoption, approve the entity dictionary, association-role semantics, quality rules and implementation-specific lifecycle contract. Maintain a change log for additions, deprecations and compatibility decisions.

CSM's scope should grow through reviewed operational needs. Its value comes from clear meanings, dependable connections and information that people can use in everyday service work.