# Critical chain: dependencies, resources and buffers

**Distinguish task dependencies from shared-resource ordering, retain an independent feeder and its protection, and identify why an un-timed network cannot establish a promised completion date.**

## Two ready tasks still need one specialist

Fictional engineering project: tasks A and B can both start after the project is ready, but each needs the same specialist. Integration requires outputs from A, B and an independent feeder F performed by another resource. Planner Hana must turn the logical network into a resource-feasible proposal. Durations and buffer sizes are not supplied.

## Your role and the reference condition

Project planner with task owners and the shared specialist.
Normal: The plan respects prerequisites, finite resource availability and the accepted project outcome.
Gap: A schedule starts A and B simultaneously and omits the feeder's joining risk.

## Work within these conditions

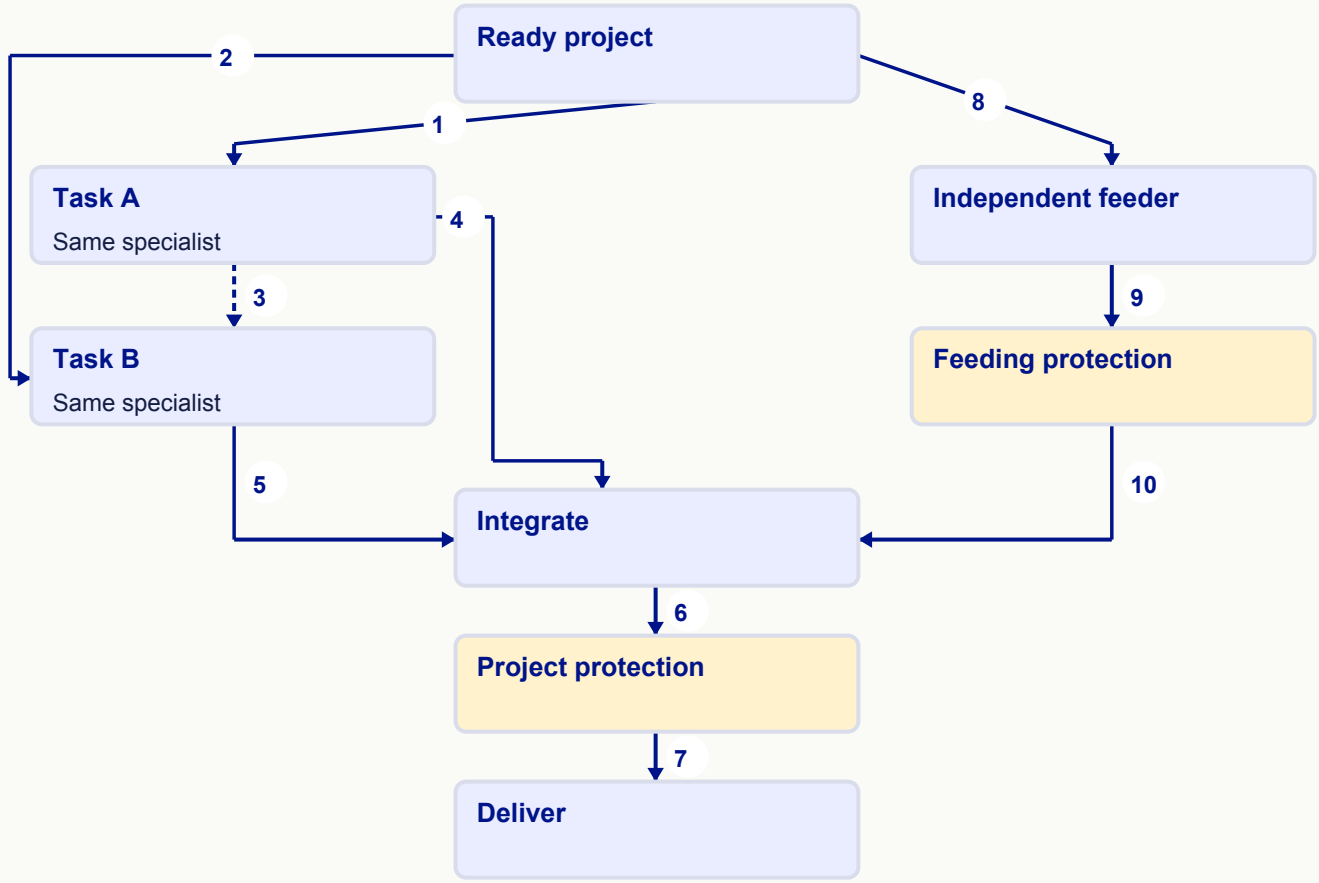
1. A-before-B is the illustrative resource ordering, not a claim that B technically depends on A.

2. No durations are supplied; the example cannot calculate a critical chain length or justified buffer size.

Fictional teaching case. Supplied observations support the exercise; proposed actions are not verified customer results.

# See how the method works

Project delay can come from resource contention even when a task-dependency diagram looks reasonable. Identify the shared resources as well as the order in which work must occur, then build a feasible plan. Make uncertainty protection visible at the project and feeding connections rather than hiding it independently in every task. Limit harmful switching among too many active jobs and review buffer consumption to focus support. Do not mechanically halve all estimates or label an ordinary critical path a critical chain. The small network introduces the mechanism; actual project commitments require credible estimates, resource calendars and the relevant planning expertise.




**1. Ready project > Task A**
Task dependency

**2. Ready project > Task B**
Task dependency

**3. Task A > Task B**
Resource ordering needed

**4. Task A > Integrate**
Required output

**5. Task B > Integrate**
Required output

**6. Integrate > Project protection**
Protect commitment

**7. Project protection > Deliver**
Delivery target

**8. Ready project > Independent feeder**
Feeder dependency

**9. Independent feeder > Feeding protection**
Protect connection

**10. Feeding protection > Integrate**
Required feeder output

One shared specialist; independent feeder resource. No task durations or buffer-sizing formula supplied.

The project network distinguishes task dependencies, a shared-resource ordering link, an independent feeder and protection points. Durations are not supplied.

# Follow the evidence and decisions

## Inspect the supplied case inputs

| Task / protection | Dependency and resource |
|---|---|
| A | Ready project; shared specialist |
| B | Ready project; same specialist |
| F: independent feeder | Ready project; separate resource |
| Integration | Requires outputs A, B and F |
| Feeding protection | Protect the feeder's join under justified policy |
| Project protection | Protect accepted delivery under justified policy |

### 01 / Draw the logical dependencies

Hana connects Ready to A, B and F, then connects all three required outputs to Integration. She records acceptance conditions for each handoff.

Why: A missing feeder can make an apparently complete main path unusable. Logical readiness and resource availability are separate questions.

Evidence / check: Integration has three prerequisite outputs, including F.

### 02 / Resolve the shared specialist conflict

For the illustrative proposal, sequence A before B on the specialist's lane. Label that link as resource ordering, not a newly discovered technical dependency.

Why: One specialist cannot perform incompatible tasks simultaneously. Distinguishing link types preserves options if priorities or resource availability change.

Evidence / check: A and B no longer overlap on the same resource; F can proceed independently if its own conditions hold.

### 03 / Locate protection without inventing its size

Show feeding protection at F's connection into Integration and project protection before committed accepted delivery. Leave durations and sizing decisions unspecified.

Why: Protective buffers are explicit management provisions. Their placement and sizing need a coherent project model and uncertainty evidence, not decorative boxes or a universal percentage.

Evidence / check: Feeding and project protection have different locations and purposes; no numeric sizes are claimed.

### 04 / Obtain the data required for a real schedule

Gather task-duration estimates with their basis, calendars, resource commitments, readiness constraints and risk assumptions. Then analyze a resource-feasible network and justified protection policy.

Why: Without durations and resource conditions, the drawing teaches interaction but cannot identify the actual limiting chain or promise a completion date.

Evidence / check: Required scheduling inputs and responsible owners are listed.

### 05 / Manage execution and learn from protection use

During the eventual approved plan, keep readiness, remaining work and agreed priorities visible. Review repeated interruptions or feeder failures with their owners.

Why: Starting many tasks to show activity can increase harmful switching while delaying the accepted project outcome.

Evidence / check: Execution review addresses the project's commitments rather than each task's appearance of busyness.

# Completed network audit - no invented durations

| Link / field | Classification | Audit conclusion |
|---|---|---|
| Ready > A and B | Task prerequisites | Both logically ready |
| A > B in proposal | Resource ordering | Shared specialist cannot overlap |
| F > Integration | Independent feeder dependency | Must remain visible |
| Feeding / project protection | Distinct provisions | Sizing evidence required |
| Completion date / chain length | Not calculable here | Durations and calendars absent |

### The feeder now needs the same specialist

A scope change makes F depend on the same specialist as A and B.

### Decision

Rebuild the resource-feasible sequence and reassess the network/protection assumptions; do not leave F running independently on the old chart.

### Reasoning

Changing a resource requirement can change the limiting schedule even when the logical output dependencies remain unchanged.

### What would change the decision?

Revised resource assignment and sequence are explicit; timing remains uncalculated until data exist.

# Practise with a changed case

## An integration project with a different join

Separate fictional project: C and D are logically independent and need one test technician. E needs a separate analyst. Final review requires C, D and E. Policy gives C priority before D. No durations are provided.

| Task | Resource / dependency |
|---|---|
| C | Technician; project ready |
| D | Same technician; project ready |
| E | Separate analyst; project ready |
| Final review | Requires C, D and E |
| Priority | C before D |

## Your task

1. Draw logical links and the distinct resource-order link.
2. Show where an independent E feeder joins and where protection decisions belong.
3. List what is missing before a delivery date or buffer size can be calculated.

Use the next page to record your reasoning before reading the answer.

# Your practice worksheet

## An integration project with a different join

### Logical prerequisites

### Resource lane and ordering

### Feeder join

### Project/feeding protection purpose

### Missing timing and uncertainty basis

Complete your reasoning before turning to the answer. Use the separate learner workbook for group practice.

# Check your reasoning and transfer it

1. C-before-D is the supplied resource order; both remain logically available after project readiness. E can proceed on its separate resource under its own conditions.

2. Final review must wait for all three outputs. Feeder protection concerns E's join; project protection concerns the accepted project commitment.

3. Durations, calendars, resource commitments and uncertainty/sizing policy are missing. A numerical critical chain or promise would be invented.

## Suggested completed record

| Network element | Meaning | Decision |
|---|---|---|
| C > D | Resource order | One technician |
| C/D/E > review | Output dependencies | All three required |
| E feeder | Separate analyst | Protect its joining condition |
| Numeric schedule | Inputs missing | No calculated date or size |

## Plausible wrong turns

1. A resource link is not always a technical prerequisite.

2. A named buffer is not a justified duration.

## Evidence of a sound answer

1. Keep the independent feeder visible.

2. Avoid double-booking the specialist.

3. State the missing scheduling evidence.

## Take the learning back to work

| Commitment | Agreed follow-through |
|---|---|
| Owner | Project owner with planner and resource owners |
| Record | Resource-feasible network, estimate basis and explicit protection policy |
| Review | At planning approval and meaningful scope/resource changes, then during execution |
| Verification evidence | Feasible assignments, ready handoffs and accepted project delivery against the approved basis |
| If unresolved | Replan through the project owner; address recurring readiness or resource conflicts rather than hiding task padding. |

# Help someone else apply the method

## Prepare

1. Task cards, two resource lanes and two distinct protection symbols
2. Blank network; no fabricated time grid

| Minutes | Activity and coaching prompt |
|---|---|
| 4 | Separate two kinds of readiness<br>Can both tasks be ready while only one can run? |
| 8 | Build the A/B/F network<br>Which link is caused only by the shared specialist? |
| 8 | Solve the C/D/E case<br>What makes the feeder independent? |
| 5 | Challenge a promised date<br>Which missing inputs make that promise unsupported? |

## Debrief

1. Ask learners to redraw the feeder after the resource-change exception.
2. Do not accept a numeric buffer inserted without a basis.

## Individual or remote practice

Use different line styles for prerequisites and resource order, then audit every integration input.

# Connect the method and check its boundaries

## Before application

1. Credible task estimates
2. Resource availability

## Outside this lesson

1. Critical chain is not just critical path renamed
2. Do not mechanically halve every estimate

## TOCICO: Introduction to Critical Chain

TOCICO: Introduction to Critical Chain

Resource dependencies, aggregated project/feeding buffers and harmful multitasking; public synopsis only

Public course synopsis only; no claim to have viewed restricted course content.

Open this lesson and its visual aids