RADIANT
AGILITY

TECHNOLOGY

Your Partner in Agile Excellence

🗺️ Epic Dependency Mapping Template

Visualize, Plan, and Execute Complex Story Dependencies

From Radiant Agility Technology

📋 How to Use This Template

1. Fill out the Story Inventory with your epic's stories
2. Complete the Dependency Matrix to identify relationships
3. Use the Sprint Planning Guide to sequence your work
4. Apply Risk Mitigation strategies for high-risk dependencies
5. Track with the Progress Dashboard throughout execution

📊 Story Inventory Worksheet

Fill this out for each story in your epic:

Story #1

STORY ID	**TITLE**
[S1]	[Story title]
DESCRIPTION	**STORY POINTS**
[Brief 1-2 sentence description]	[1, 2, 3, 5, 8]
BUSINESS VALUE	**TECHNICAL RISK**
[High/Medium/Low]	[High/Medium/Low]
ASSIGNED TEAM	
[Team name]	

Story #2

STORY ID	**TITLE**
[S2]	[Story title]

DESCRIPTION	**STORY POINTS**
[Brief 1-2 sentence description]	[1, 2, 3, 5, 8]
BUSINESS VALUE	**TECHNICAL RISK**
[High/Medium/Low]	[High/Medium/Low]
ASSIGNED TEAM	
[Team name]	

Story #3

STORY ID	**TITLE**
[S3]	[Story title]
DESCRIPTION	**STORY POINTS**
[Brief 1-2 sentence description]	[1, 2, 3, 5, 8]
BUSINESS VALUE	**TECHNICAL RISK**
[High/Medium/Low]	[High/Medium/Low]
ASSIGNED TEAM	
[Team name]	

Continue for all stories in your epic...

🔗 Dependency Matrix Template

Mark the relationship between stories using this matrix:

Legend:

- ✅ **Must Complete First** - Story A must be 100% done before Story B can start
- 🔄 **Partial Dependency** - Story A needs to be partially complete (specific criteria met)
- ⚠️ **Soft Dependency** - Story A should ideally be done first, but Story B can start in parallel
- 🚫 **No Dependency** - Stories can be worked completely independently

Dependency Matrix	S1	S2	S3	S4	S5	S6
S1	-					
S2		-				
S3			-			
S4				-		
S5					-	

Dependency Matrix	S1	S2	S3	S4	S5	S6
S6						-
S7						
S8						

Dependency Details

For each dependency marked above, provide details:

S[X] → S[Y] Dependency:

TYPE	SPECIFIC REQUIREMENT
[Must Complete First/ Partial/Soft]	[What exactly must be delivered?]
IMPACT IF BLOCKED	**WORKAROUND AVAILABLE**
[What happens if S[X] is delayed?]	[Yes/No - describe if yes]

🛤️ Critical Path Analysis

Critical Path Identification

List the longest chain of dependent stories (this determines your minimum timeline):

Critical Path:

S__ → S__ → S__ → S__

TOTAL STORY POINTS

```
[Sum of critical path stories]
```

ESTIMATED DURATION

```
[Based on team velocity]
```

BUFFER NEEDED

```
[Recommended 20-30% buffer]
```

Parallel Work Streams

Identify stories that can be worked simultaneously:

STREAM A (CRITICAL PATH)

```
S__ → S__ → S__
```

STREAM B (PARALLEL)

```
S__ → S__
```

STREAM C (INDEPENDENT)

```
S__
```

Bottleneck Analysis

Identify potential bottlenecks:

Story S__ is a bottleneck because:

REQUIRED BY STORIES	TECHNICAL COMPLEXITY
[List dependent stories]	[High/Medium/Low]
RESOURCE CONSTRAINTS	MITIGATION STRATEGY
[Specialized skills needed]	[Your approach to address this bottleneck]

📅 Sprint Planning Framework

Sprint 1 Plan

Goal: [Sprint goal focused on early value delivery]

Stories to Include:

- ☐ S__ (_ points) - [Brief description and rationale]
- ☐ S__ (_ points) - [Brief description and rationale]
- ☐ S__ (_ points) - [Brief description and rationale]

TOTAL POINTS	TEAM CAPACITY
___	[Your team's average velocity]

RISK LEVEL

[Low/Medium/High]

Success Criteria:

- ☐ [Specific deliverable 1]
- ☐ [Specific deliverable 2]
- ☐ [Key milestone achieved]

Sprint 2 Plan

Goal: [Next sprint goal building on Sprint 1]

Stories to Include:

- ☐ S__ (_ points) - [Brief description and dependencies]
- ☐ S__ (_ points) - [Brief description and dependencies]

Dependencies to Verify:

- ☐ [Specific dependency from Sprint 1]
- ☐ [Integration point confirmed]
- ☐ [Required output available]

Risk Mitigation:

- ☐ [Backup plan if dependencies delayed]
- ☐ [Alternative approach identified]

Sprint 3 Plan

Goal: [Final sprint goal to complete epic]

Stories to Include:

- ☐ S__ (_ points) - [Brief description]
- ☐ S__ (_ points) - [Brief description]

Integration Points:

- ☐ [Cross-story integration testing]
- ☐ [End-to-end validation]
- ☐ [Epic completion verification]

⚠️ Risk Assessment & Mitigation

High-Risk Dependencies

Risk #1: [Describe the dependency risk]

RISK LEVEL	IMPACT	PROBABILITY
High/ Medium/Low	[What happens if this dependency fails]	[Likelihood of the risk occurring]

Mitigation Strategy:

- ☐ [Specific action 1]
- ☐ [Specific action 2]
- ☐ [Backup plan]
- ☐ [Escalation path]

Risk #2: [Another dependency risk]

RISK LEVEL	IMPACT	PROBABILITY
High/ Medium/Low	[Impact description]	[Likelihood assessment]

Mitigation Strategy:

- ☐ [Mitigation action 1]

- ☐ [Mitigation action 2]

Dependency Workarounds

If S[X] is Delayed:

- ☐ Option A: [Alternative approach]
- ☐ Option B: [Workaround solution]
- ☐ Option C: [Scope adjustment]

If S[Y] is Blocked:

- ☐ Option A: [Parallel work option]
- ☐ Option B: [Mock/stub solution]
- ☐ Option C: [Priority resequencing]

📊 Dependency Health Dashboard

Current Status Overview

Story	Status	Dependencies	Blocker	Risk Level	ETA
S1	[Status]	[Dependencies]	[Current blocker]	[Risk]	[Date]
S2	[Status]	[Dependencies]	[Current blocker]	[Risk]	[Date]
S3	[Status]	[Dependencies]	[Current blocker]	[Risk]	[Date]
S4	[Status]	[Dependencies]	[Current blocker]	[Risk]	[Date]
S5	[Status]	[Dependencies]	[Current blocker]	[Risk]	[Date]

Dependency Health Indicators

● **Green (Healthy):**

- All dependencies on track
- No blocking issues identified
- Timeline achievable with current progress

● **Yellow (Watch Closely):**

- Minor delays possible but manageable
- Workarounds available if needed

- May impact sprint goal but not epic completion

🔴 Red (Action Required):

- Major blocker identified requiring immediate attention
- Timeline at significant risk
- Escalation and intervention needed

CURRENT OVERALL STATUS

```
[🟢/🟡/🟥]
```

REASON

```
[Brief explanation of current situation]
```

🔄 Weekly Dependency Review Process

Monday Check-in Questions:

1. Are all dependencies marked as complete actually done and verified?
2. Have any new dependencies been discovered since last week?
3. Are there any stories at risk of becoming blockers?
4. What support do teams need to meet their commitments?

Wednesday Status Update:

1. Update the dependency health dashboard
2. Communicate any status changes to affected teams
3. Adjust sprint plans if necessary based on new information
4. Identify any escalations needed for Friday's planning

Friday Planning Review:

1. Review next week's planned work against current dependencies
2. Confirm all prerequisite work will be completed on time
3. Adjust priorities if dependencies have shifted
4. Plan mitigation actions for any identified risks

🎯 Dependency Types Reference Guide

Technical Dependencies

API Dependencies:

- Story A must expose API before Story B can consume it
- **Planning tip:** Define API contract early, build with mocks first

Database Dependencies:

- Story A must create/migrate schema before Story B can use it
- **Planning tip:** Separate schema changes from business logic

Infrastructure Dependencies:

- Story A must set up environment before Story B can deploy
- **Planning tip:** Infrastructure as Code, parallel environment setup

Business Dependencies

User Experience Flow:

- Users must complete Step A before Step B makes sense
- **Planning tip:** Can we provide value at each step independently?

Content Dependencies:

- Story A must define content structure before Story B can display it
- **Planning tip:** Create content templates/wireframes early

Compliance Dependencies:

- Story A must implement security before Story B can handle sensitive data
- **Planning tip:** Security reviews early in development

Resource Dependencies

Shared Team Members:

- Both stories need the same specialist
- **Planning tip:** Time-box specialist involvement, cross-train team

External Dependencies:

- Stories need third-party approvals or integrations
- **Planning tip:** Start external processes early, have fallback plans

📈 Success Metrics for Dependency Management

Leading Indicators (Track Weekly):

Dependency prediction accuracy	% identified vs discovered
Dependency resolution time	Avg time to resolution
Blocked story percentage	% currently blocked
Cross-team communication	Discussions per week

Lagging Indicators (Track Monthly):

Sprint goal achievement	% goals delivered
Epic completion predictability	Planned vs actual dates
Rework due to dependencies	Amount of rework
Team satisfaction	Process rating

Target Benchmarks:

🎯 Excellent:

>90% dependency prediction, <1 day resolution time, <10% blocked stories

✅ Good:

80-90% prediction, 1-3 day resolution, 10-20% blocked stories

⚠️ Needs Improvement:

70-80% prediction, 3-7 day resolution, 20-30% blocked stories

🚨 **Critical:**

<70% prediction, >7 day resolution, >30% blocked stories

🛠️ Tool Integration Guide

For Jira Teams:

Custom Fields to Add:

- "Depends On" (Issue Link)
- "Dependency Type" (Must Complete/Partial/Soft)
- "Blocker Impact" (High/Medium/Low)
- "Workaround Available" (Yes/No)

Useful JQL Queries:

```
# All blocked stories status != Done AND "Depends
On" is not empty AND "Depends On" not in (Done) #
High-risk blocked stories status != Done AND
"Blocker Impact" = High AND "Depends On" not in
(Done) # Stories ready to start status = "To Do"
AND ("Depends On" is empty OR "Depends On" in
(Done))
```

For Azure DevOps Teams:

Work Item Types to Utilize:

- Use "Predecessor" and "Successor" link types
- Create custom work item rules for dependency validation
- Set up automated notifications for dependency status changes

For Notion/Linear Teams:

Database Properties:

- Relation properties to link dependent stories
- Formula properties to calculate dependency health
- Automation rules to update status based on dependencies

📞 Need Help with Complex Dependencies?

If your epic has particularly complex or risky dependencies:

● Dependency Mapping Workshops (facilitated sessions to map complex relationships)

● Cross-Team Coordination Planning (align multiple teams around shared dependencies)

● Risk Mitigation Strategy Development (create comprehensive backup plans)

● Tool Setup and Integration (configure your tools for optimal dependency tracking)

Contact: hello@radiantagility.tech

📚 Additional Resources

Related Templates:

- Epic Splitting AI Prompts Library
- Sprint Planning Checklist
- Risk Assessment Framework
- Cross-Team Communication Charter

Recommended Reading:

- "Managing Dependencies in Agile Projects" (Agile Alliance)
- "Scaled Agile Framework: Dependency Management" (SAFe Guide)
- "Team Topologies" by Matthew Skelton and Manuel Pais

Training Opportunities:

- SAFe Product Owner Certification
- Advanced Scrum Master Workshop
- Cross-Team Collaboration Masterclass

This Dependency Mapping Template is part of the Epic Splitting Toolkit by Radiant Agility Technology.
Download the complete toolkit at **radiantagility.tech/epic-toolkit**