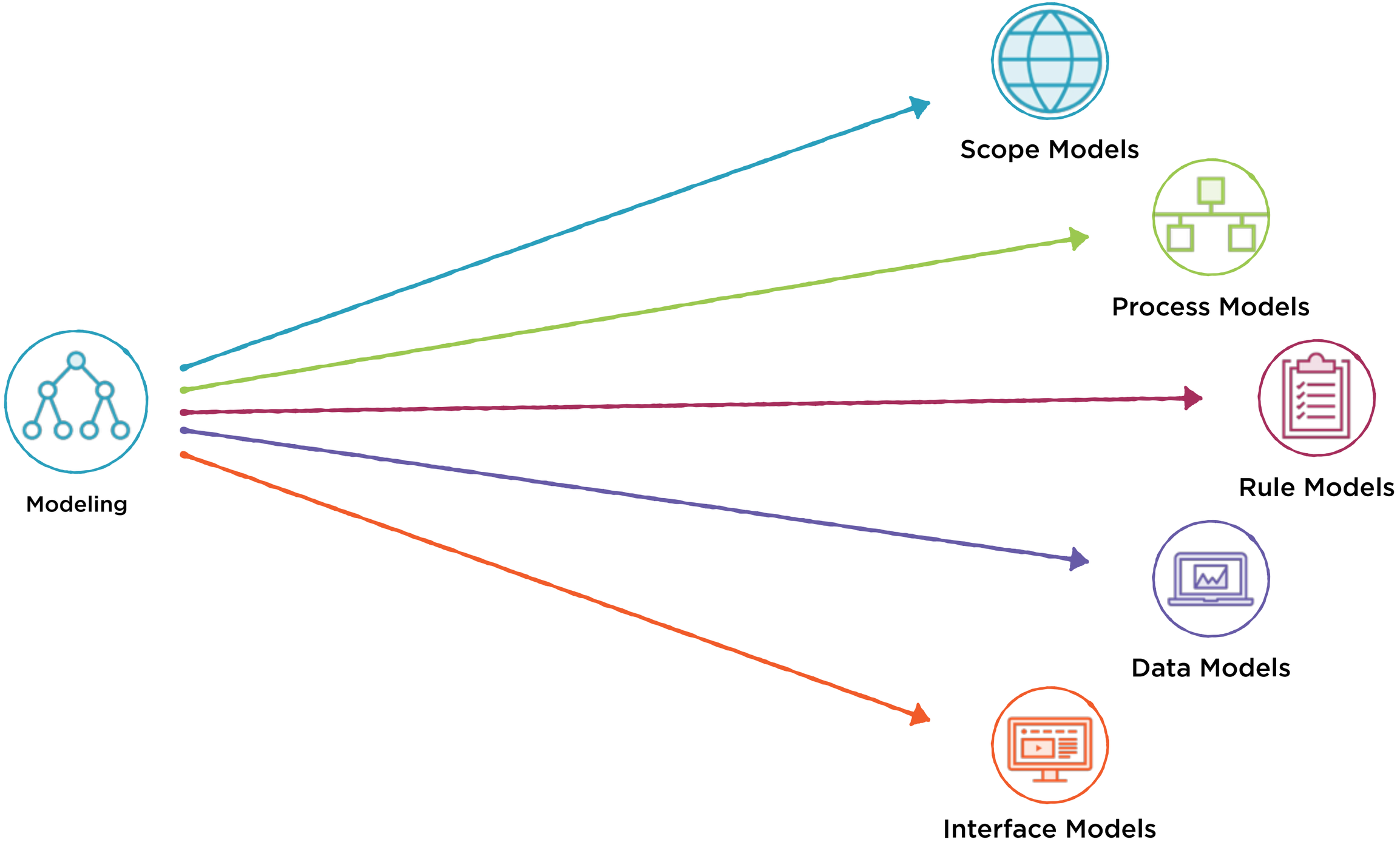
Documenting Solution Requirements

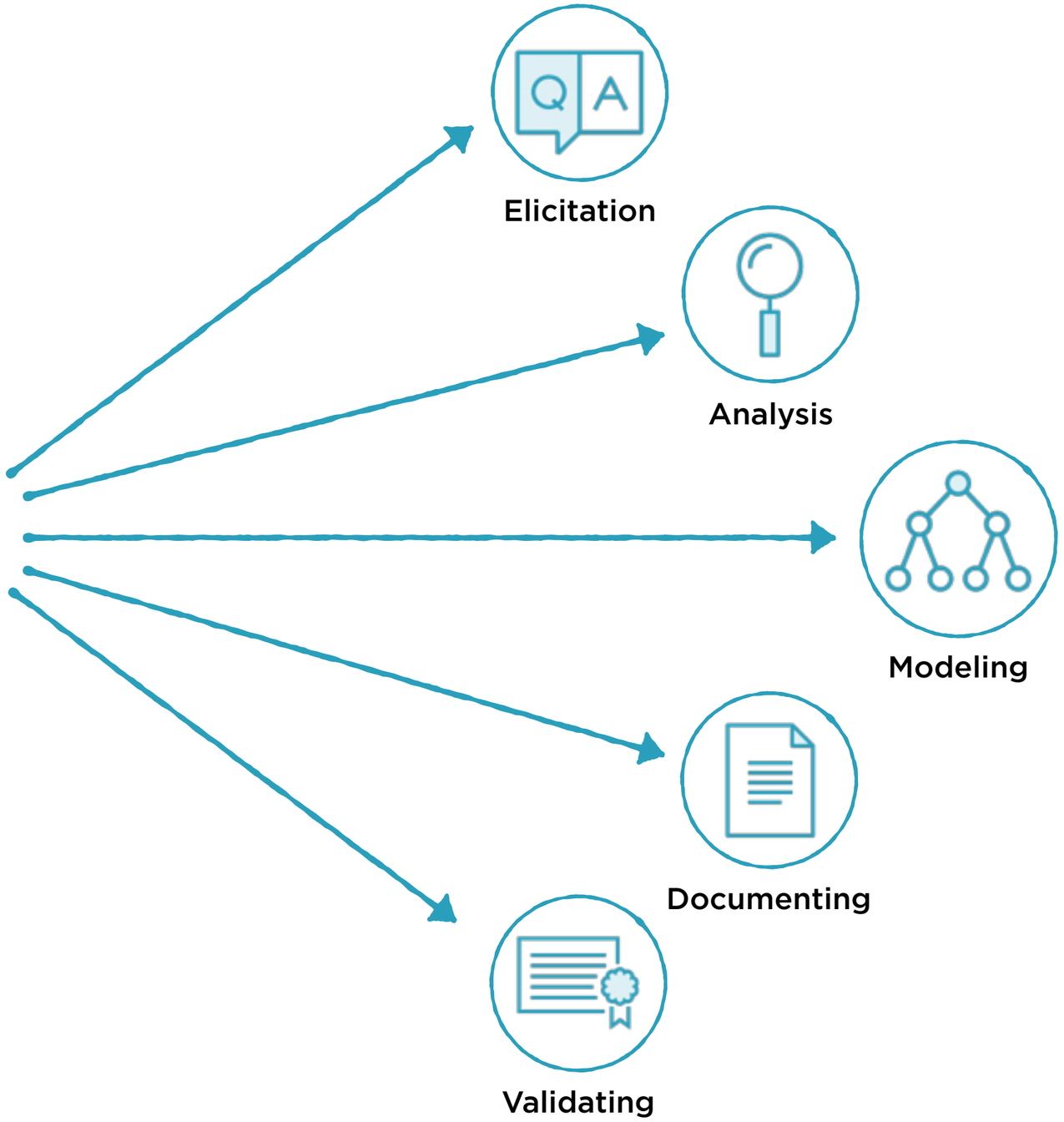
Casey Ayers

MBA • PMP®

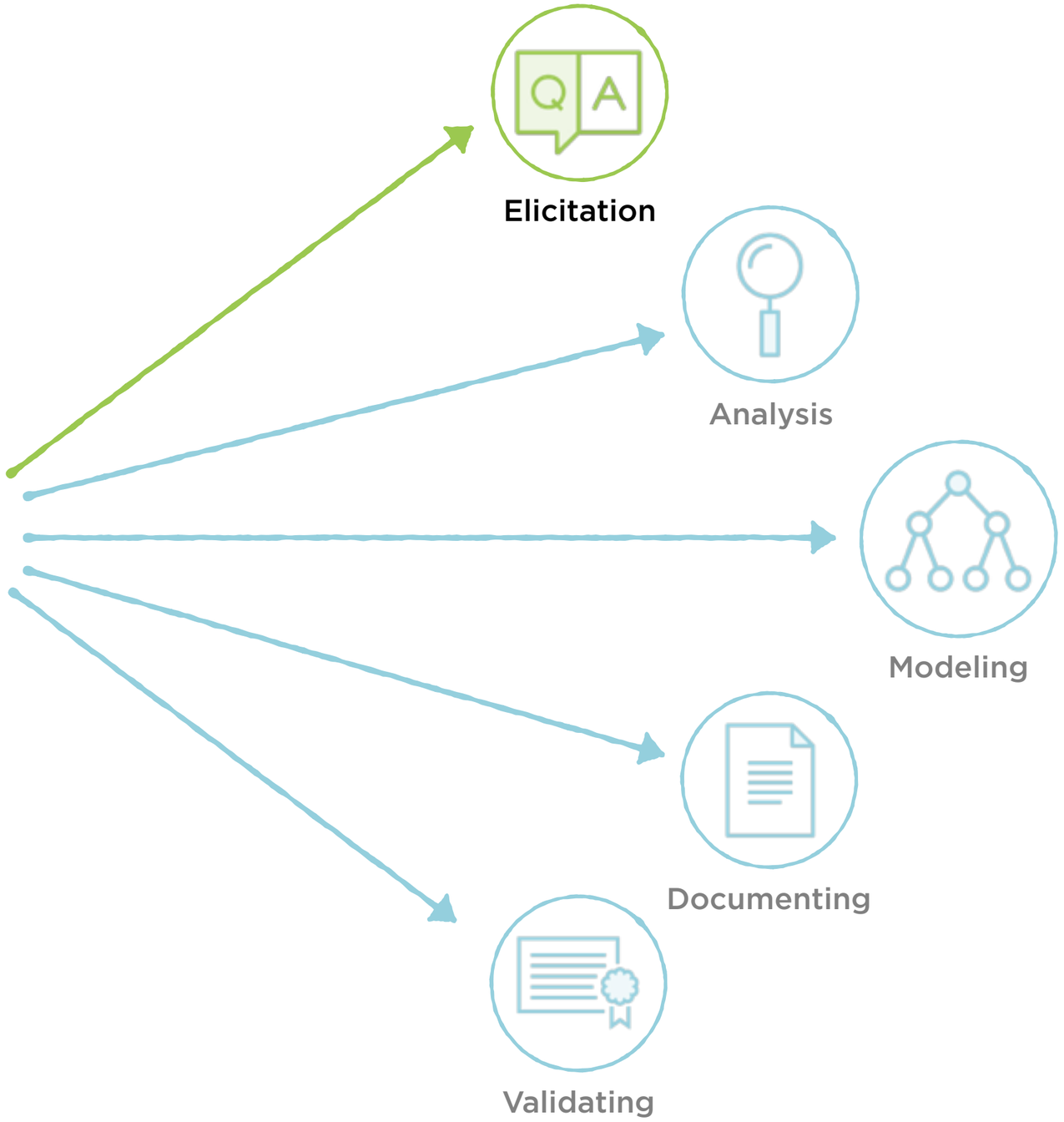
CaseyAyers.com | @caseyayers
linkedin.com/in/caseyayers


Scope Models
Process Models
Modeling
Rule Models
Data Models
Interface Models

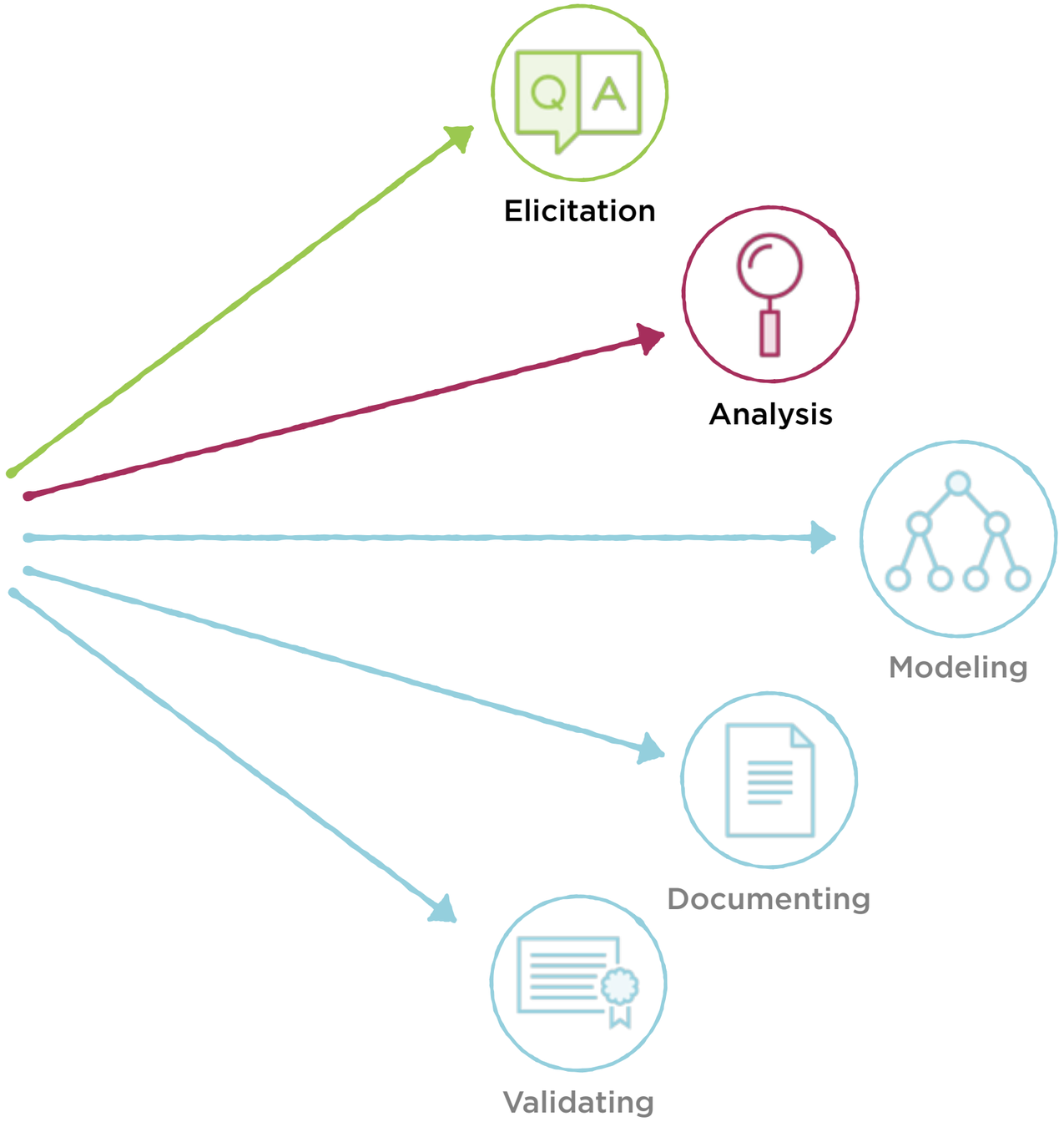
Elicit, document and manage stakeholder requirements to meet business and project objectives



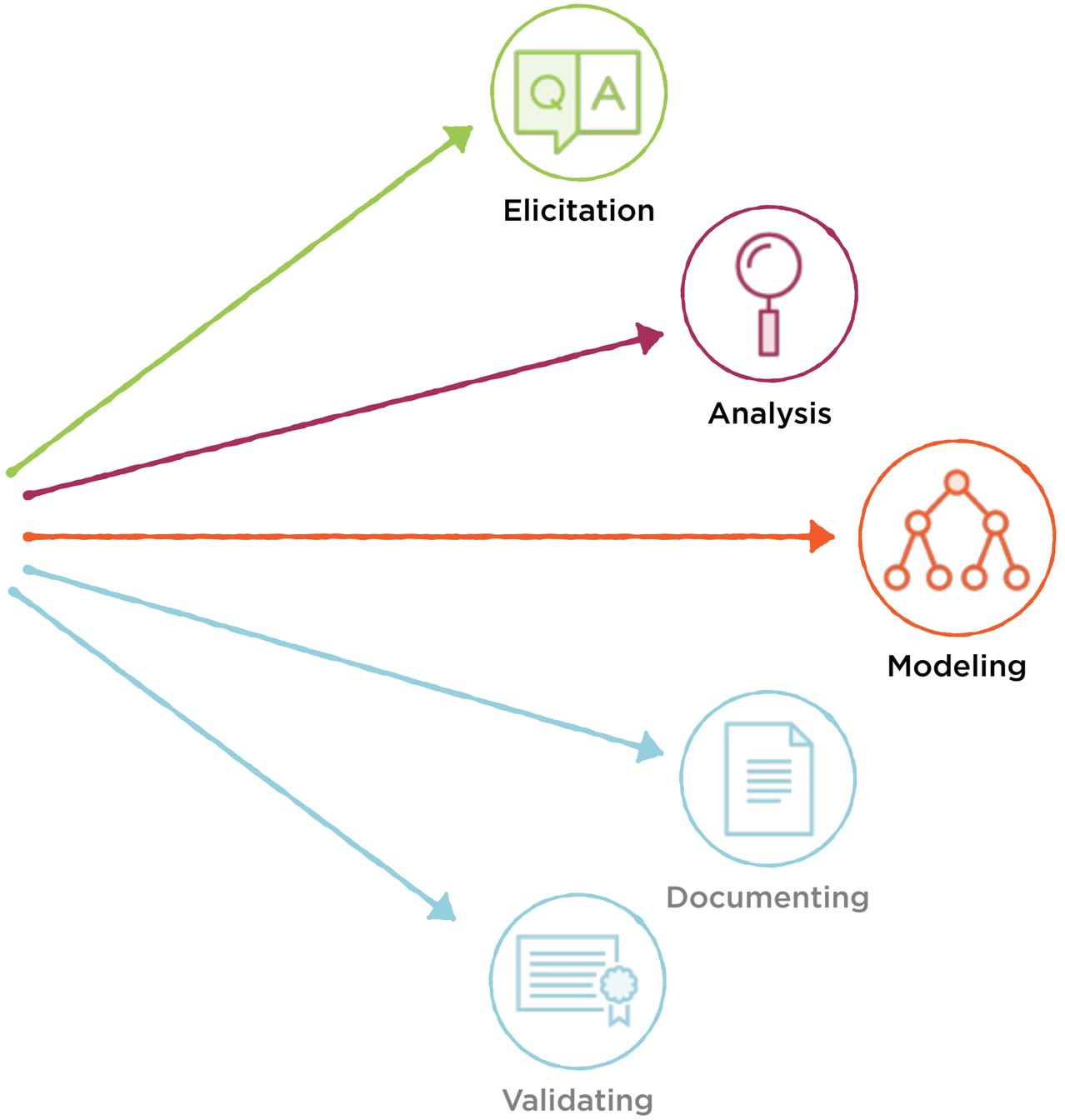
Elicit, document and manage stakeholder requirements to meet business and project objectives



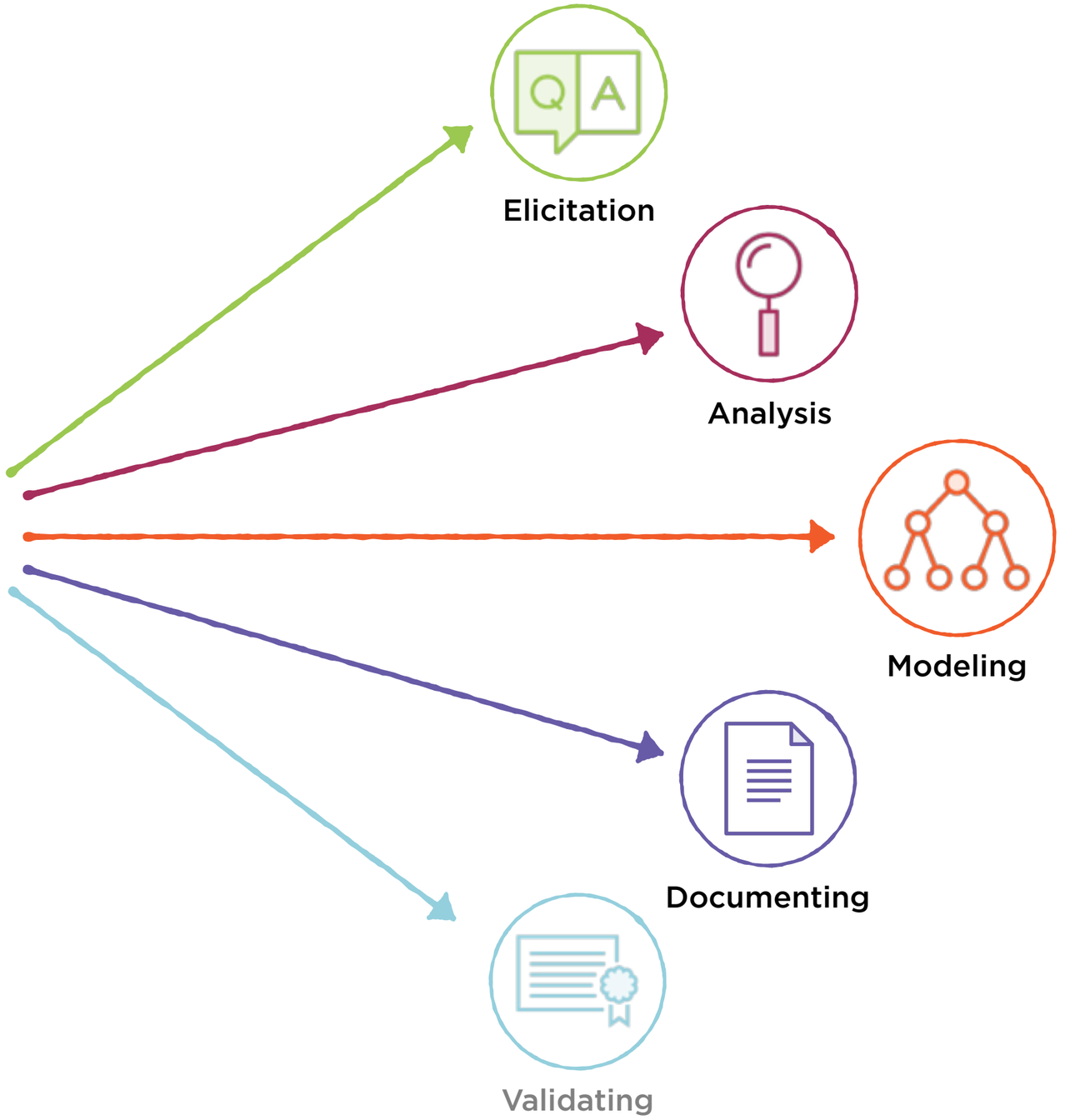
Elicit, document and manage stakeholder requirements to meet business and project objectives

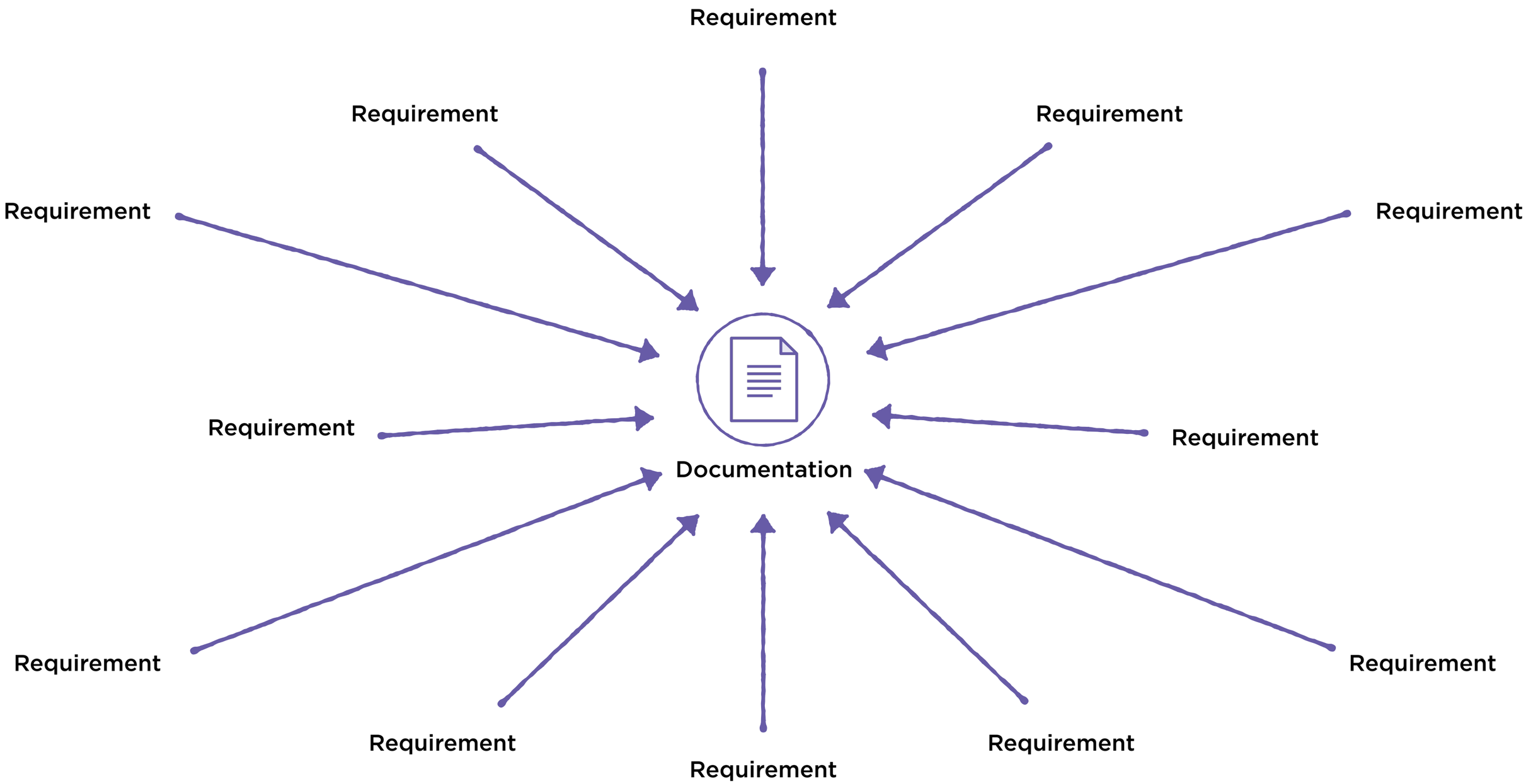


Elicit, document and manage stakeholder requirements to meet business and project objectives



Elicit, document and manage stakeholder requirements to meet business and project objectives




Requirement
Requirement
Requirement
Requirement
Requirement
Requirement
Requirement
Documentation
Requirement
Requirement
Requirement
Requirement
Requirement

Documentation

Documented requirements represent the full scope of solution to address business objectives

Solution and requirements may evolve or gain detail over time, especially in adaptive environments

Documentation

Level of formality, form of documentation varies based on project needs

Models may be used both to better understand requirements and to represent them

What's Ahead

Importance of Documentation

Documenting Solutions

Documenting Requirements

Categorizing and Filtering Requirements

What's Ahead

Documenting Assumptions & Constraints

Writing Effective Requirements

Prioritizing Requirements

Importance of Documentation

Importance of Documentation

Baseline for the Project

Outlines what qualifies as success

Allows for more genuine sign-off, greater stakeholder buy-in

Provides a reference useful in validating stakeholder needs are met

Importance of Documentation

Baseline for the Project

Traceable point from which solution may evolve to better meet needs

May serve legal, regulatory, or contractual purpose as a baseline for certain projects

Importance of Documentation

Fundamental Point of Reference

Used by designers, developers, testers, and QA staff to understand what requirements must be met

Referred to by project manager when developing project plans and schedule

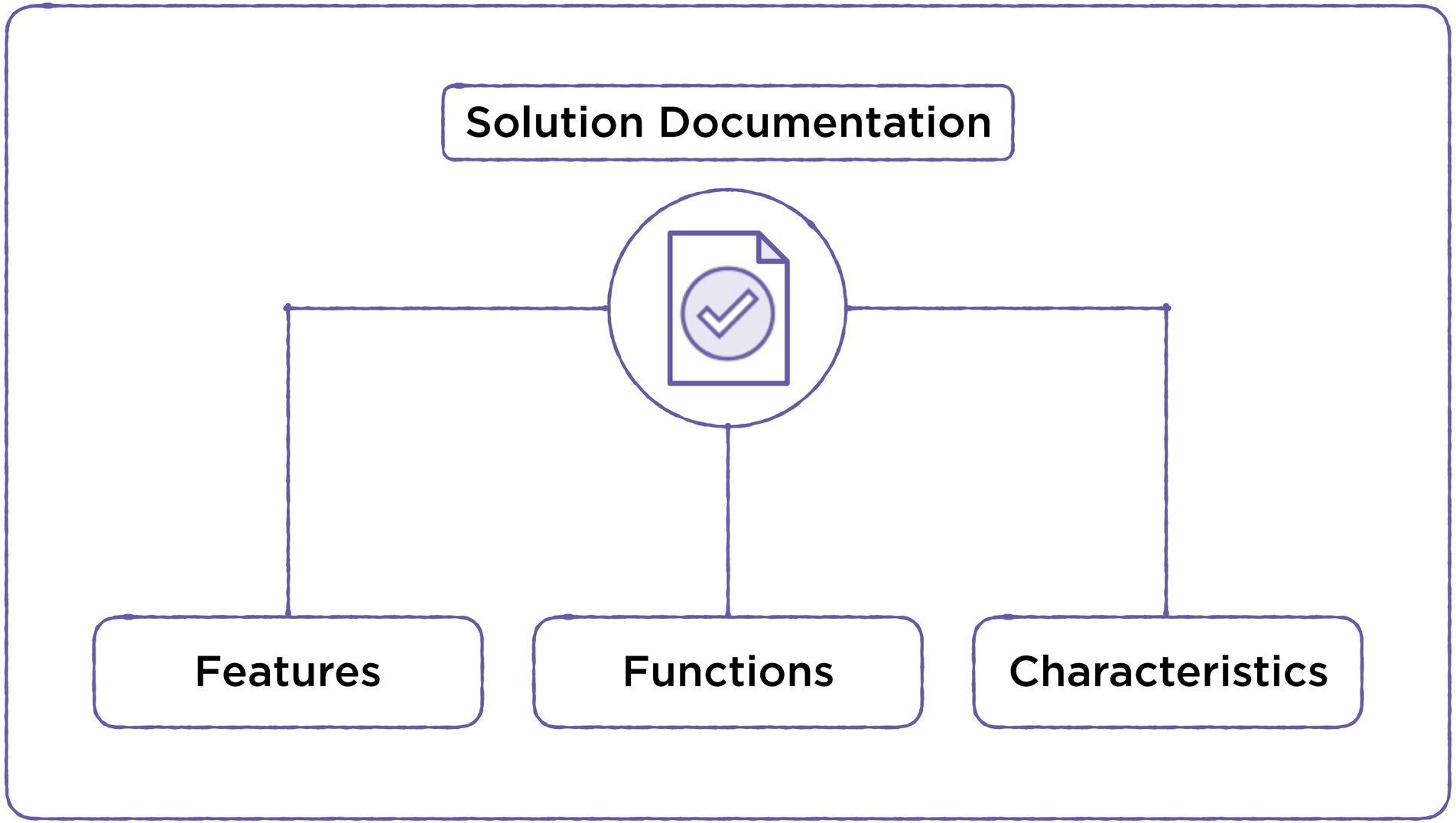
Importance of Documentation

Fundamental Point of Reference

Contractual agreements, vendor requirements typically based on documentation

Materials may be referenced or adapted when creating user documentation

Documenting Solutions

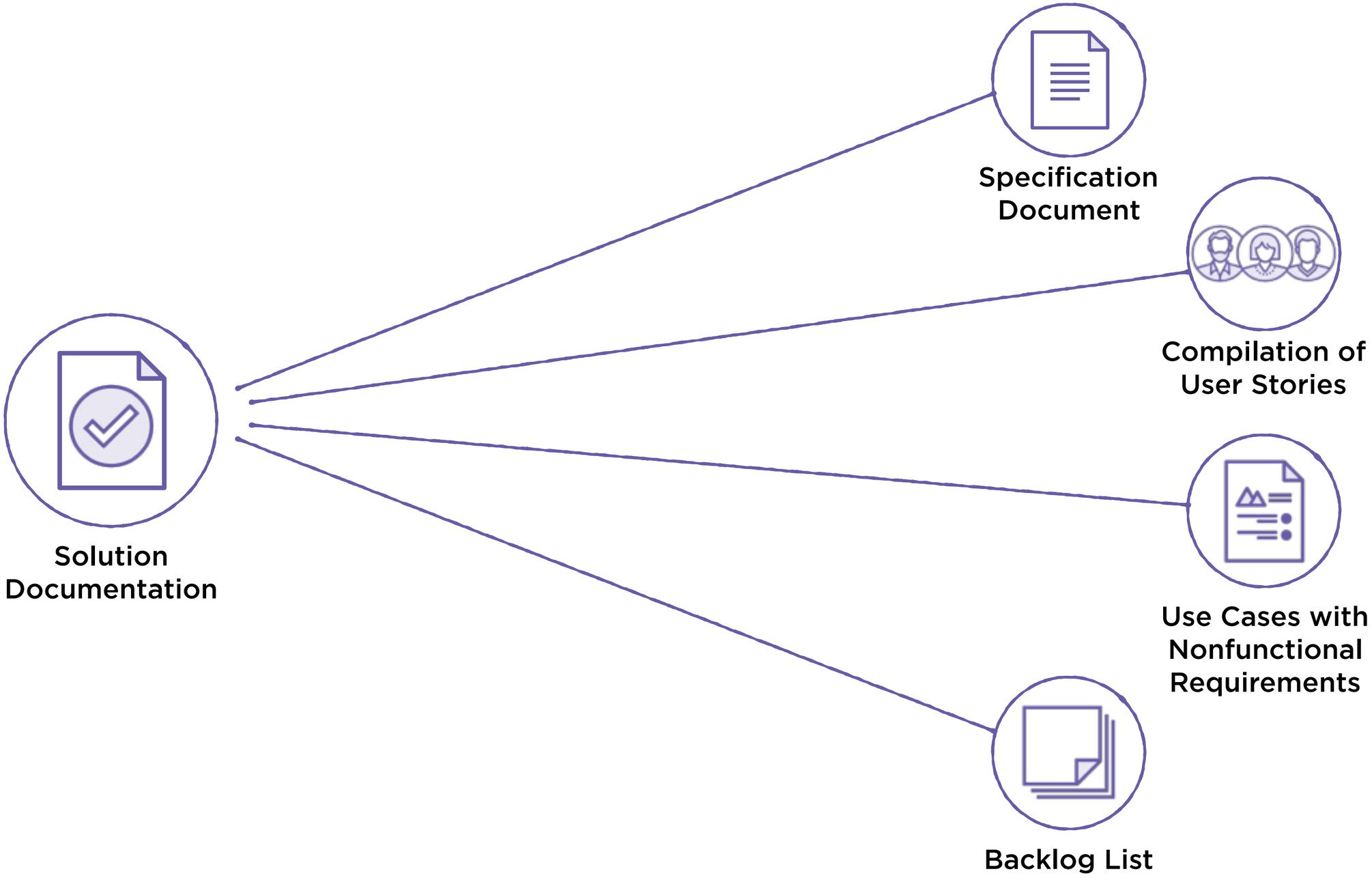

Solution Documentation
Features
Functions
Characteristics

Documenting Solutions

Implementation teams rely on solutions documentation to do their work

Especially true where vendors and outside contractors are involved

Informal, internal knowledge rarely can be relied upon in successful projects


Specification Document
Compilation of User Stories
Use Cases with Nonfunctional Requirements
Backlog List
Solution Documentation

Documenting Solutions

Format to be used should be determined during analysis planning

Best format will vary based on...

- Project scope
- Project complexity
- Organizational norms
- Audience

Documentation should capture all requirements

Documenting Solutions

Specification documentation should attempt to anticipate the...

Conditions

Actions and reactions

Results

Errors and conflicts

...that may arise as part of the solution

Documenting Solutions

The best solutions documentation describes how to create the solution in full, and how to address issues that arise

Documenting Solutions

Solution Requirements may be *functional* or *nonfunctional* in nature

Functional requirements: Describe *what* a solution does

Nonfunctional requirements: Describe *how* a solution does what it does

Documenting Solutions

Product requirements: Describe the end result that meets business needs

Project requirements: Describe mandatory conditions to create the end product

Documenting Solutions

Business requirements: Describe the high-level needs and objectives that lead to project work being undertaken

Stakeholder requirements: Describe the needs of those impacted by, or that have an impact on, the project

Documenting Solutions

Transition requirements: Describe the systems, processes, and resources necessary to successfully arrive at the desired state

Categorizing and Filtering Requirements

Categorizing Requirements

Used to make requirements easier to organize and reference

Variety of methods may be valid depending on requirements documented

Filters helpful in eliminating unnecessary or poorly structured requirements

Categorizing Requirements

Filter by scope: Verify that all documented requirements are in scope.

Remove or revise those that are out of scope or not clearly in scope

Categorizing Requirements

Filter by function: Ensure all requirements fit into a functional category

Functional categories determined based on project specifics

Categorizing Requirements

Filter by priority: Determine which requirements are most critical to meeting business and stakeholder needs

Consider removing requirements that are low priority when scope creep is an issue

Categorizing Requirements

Filter by testability: Remove requirements that cannot be independently verified or tested based on predetermined criteria

Documenting Assumptions

Documenting Assumptions

Every project and every set of requirements rests on assumptions

Identifying what is assumed, and what is fact, is key to risk management

Documenting Assumptions

Triggers for assumptions:

- Incomplete information
- Projections about future changes
- Projections about future constants

Documenting Assumptions

Assumptions that influence requirements

"We expect 102,000 ads to be served per day to players."

"We'll be profitable in six months if current growth trends continue."

"The team will be able to finish work on the new mission pack in four months."

Documenting Assumptions

Requirements influenced by assumptions

"We need enough server capacity to handle 30,000 simultaneous players."

"We need to develop three new item packs to sustain IAP revenue."

"We need to add the ability to level above Lv. 50 to the game."

Documenting Assumptions

Important to analyze requirements to determine which are based on, influenced by, or exist due to assumptions

Assumptions should be cataloged and reassessed regularly as analysis and project work proceed

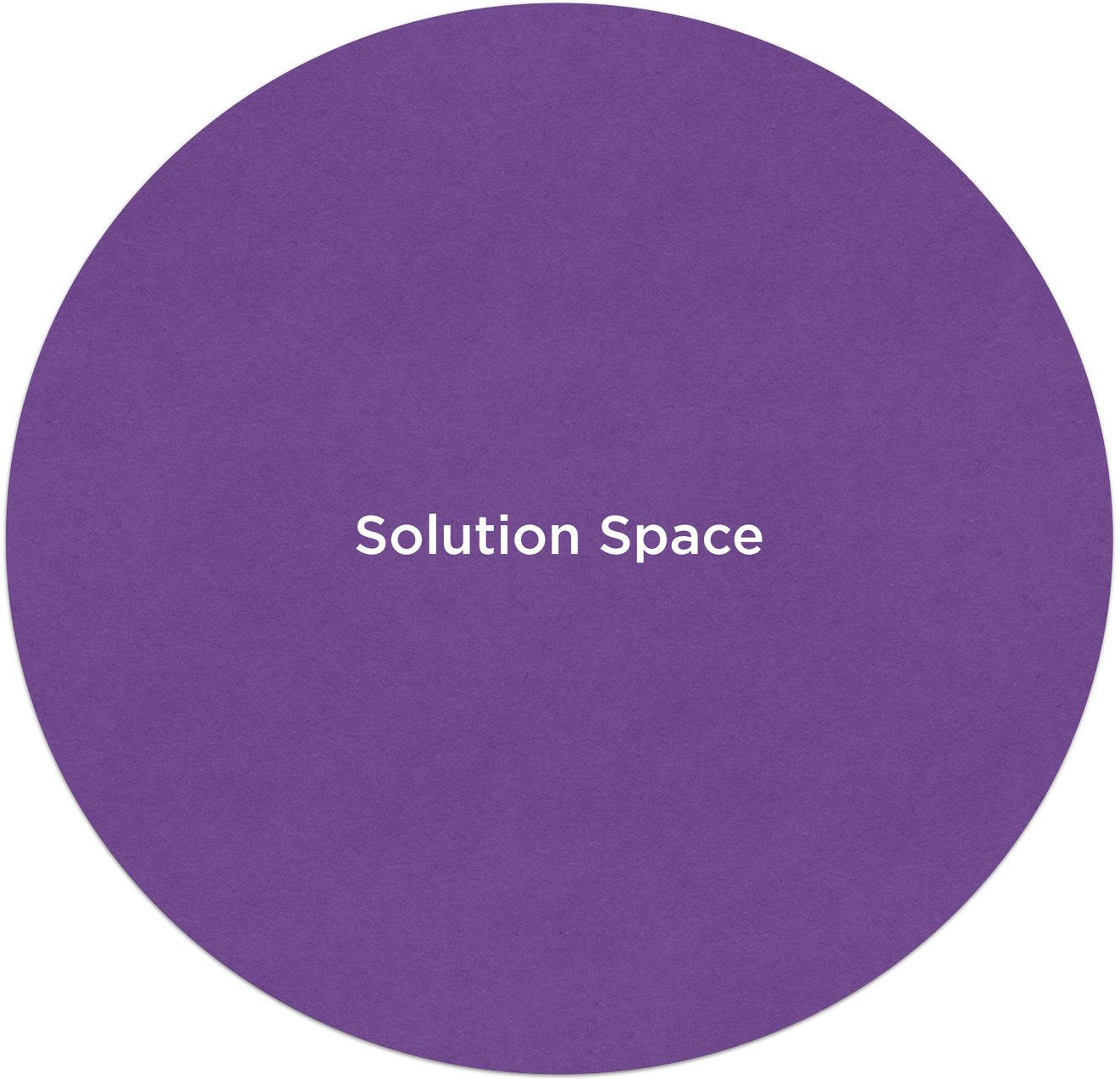
Documenting Constraints

Documenting Constraints

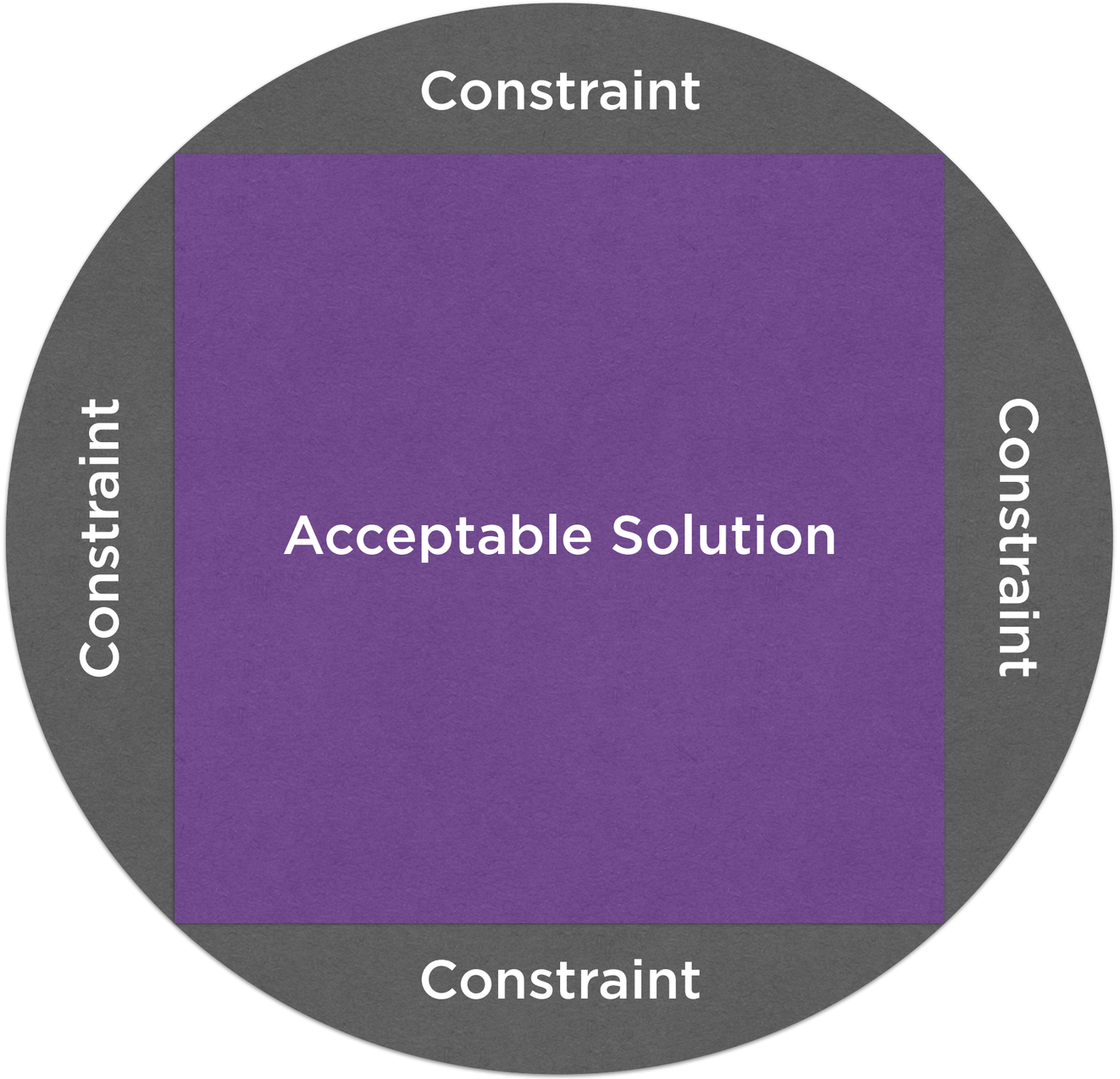
Product constraints: Limitations on how a solution can accomplish its goals

Project constraints: Limitations on how a solution can be developed

Documenting Constraints



Documenting Constraints

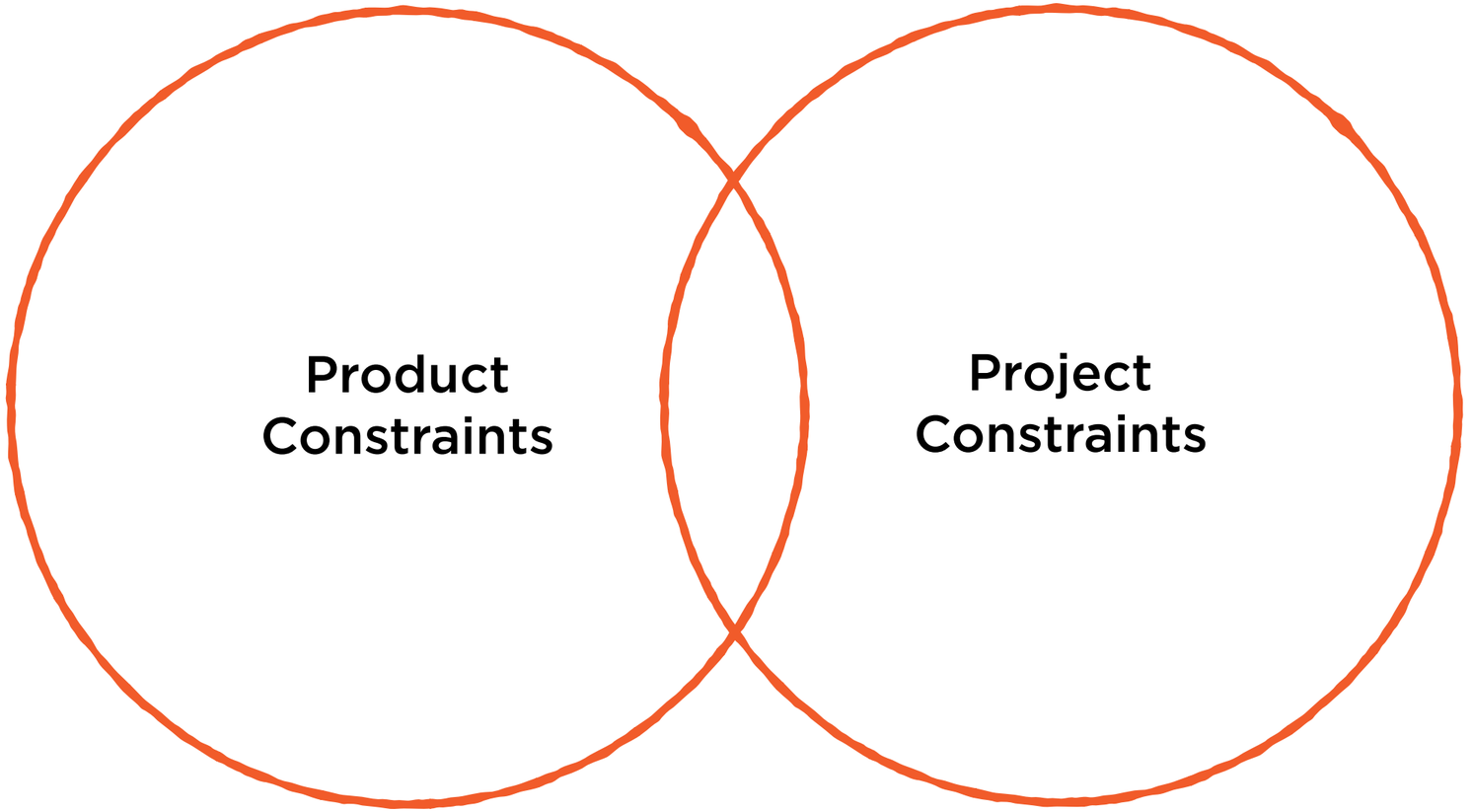


Documenting Constraints

Acceptable Solution

Project Constraints

Documenting Constraints



Documenting Constraints

Constraints often viewed as a type of nonfunctional requirement

Requirements indicate what *should* be done, while constraints indicate what *should not* be done

Documenting Constraints

Constraints may emerge from a variety of sources

Geography, organizational culture and policy, and the legal/regulatory environment may all lead to constraints

Particular attention to where assumptions and constraints intertwine can be valuable

Writing Effective Requirements

Writing Effective Requirements

All business analysis is futile if resulting requirements are unclear, conflicting, or confusing

Strongly written requirements guarantee common understanding

- Accurate baselines
- Correct implementation
- Optimal integration

Writing Effective Requirements

Well-written requirements exhibit...

- Clarity
- Precision
- Consistency
- Correctness
- Completeness
- Measurability
- Feasibility
- Traceability
- Testability

Writing Effective Requirements

Well-written requirements include...

- Condition
- Subject
- Imperative
- Active verb
- Object
- Business rule*
- Outcome*

**Optional*

Writing Effective Requirements

Sample Requirement

When the player presses the “Attack!” button, the game will present the attack screen, allowing players to select troops to deploy and powers to use to initiate a battle.

Writing Effective Requirements

When the player presses the "Attack!" button, the game will present the attack screen, allowing players to select troops and powers to use to initiate a battle.

Condition

Subject

Imperative

Active Verb

Object

Outcome

Writing Effective Requirements

Clarity

Meaning of requirement should not be subject to interpretation or debate

Precision

Requirements should use carefully select words to describe the solution specifically

Writing Effective Requirements

Gauging Clarity

The game will allow players to add people they know to a list of friends

The game will obtain social graph information from integrated social accounts, and use this data to populate a list of others the player may then add to a friends list.

Writing Effective Requirements

Gauging Clarity

The game will allow players to add people they know to a list of friends

The game will obtain social graph information from integrated social accounts, and use this data to populate a list of others the player may then add to a friends list.

Writing Effective Requirements

Gauging Precision

If players do not have an active internet connection, an error message will appear

If players do not have an active internet connection, an error message reading, “Avast! Your connection to the server is down, matey...” will appear

Writing Effective Requirements

Gauging Precision

If players do not have an active internet connection, an error message will appear

If players do not have an active internet connection, an error message reading, “Avast! Your connection to the server is down, matey...” will appear

Writing Effective Requirements

Consistency

Requirements should only be included once in a list of requirements, and should not conflict with one another

Traceability can help ensure requirements remain consistent with known information and business needs

Consistency is important, even at the risk of repetition

Writing Effective Requirements

Gauging Consistency

- Missions
- Battles
- Conquests
- Incursions

Writing Effective Requirements

Gauging Consistency

- Missions
- Missions
- Missions
- Missions

Writing Effective Requirements

Consistency

Referencing back to requirements, rather than rewriting them, is one way to help ensure consistency

Similar to use of pointers and variables in programming

Writing Effective Requirements

Correctness

Requirements should align with known facts and running assumptions

Should facts or assumptions change, important to revise requirements

All requirements should be vetted and validated by stakeholders with relevant knowledge

Writing Effective Requirements

Completeness

Not all information often known when requirements are written

Write requirements that actively acknowledge this, rather than obscure it

Good requirements are self-contained, including all known and unknown factors

Writing Effective Requirements

Gauging Completeness

Approved new items will be added to the game marketplace

All items approved (number TBD) by the Content Director by the deadline (TBD) will be added to the game marketplace

Writing Effective Requirements

Gauging Completeness

Approved new items will be added to the game marketplace

All items approved (number TBD) by the Content Director by the deadline (TBD) will be added to the game marketplace

Writing Effective Requirements

Measurability

Requirements cannot be verified if they cannot be measured

Measurements should be possible independent of other requirements

Writing Effective Requirements

Gauging Measurability

Updates will lead to an increase in average session play time

Updates will lead to an increase in average session play time by 10-20%

Writing Effective Requirements

Gauging Measurability

Updates will lead to an increase in average session play time

Updates will lead to an increase in average session play time by 10-20%

Writing Effective Requirements

Feasibility

Requirements that cannot be met harm long-term buy-in and impede progress

Important to ensure requirements are realistic given...

- Organizational capabilities
- Available resources
- Risk tolerance
- Level of commitment

Writing Effective Requirements

Feasibility

Operational: Will solution have necessary stakeholder support and adoption to succeed?

Technical: Can the requirement actually be met with available technology?

Cost-effectiveness: Is the requirement worth meeting given the cost vs. benefit?

Time: Can the solution be created in a useful and reasonable amount of time?

Writing Effective Requirements

Traceability

Requirements that can’t be traced back to business needs are invalid

Changes and new information may often render some requirements invalid, or result in a revision being necessary

Writing Effective Requirements

Testability

Similar to measurability in most cases

Some requirements may be subjected to a pass/fail examination, rather than be measured on a more descriptive scale

Testability must be confirmed, but a test need not necessarily be made upfront

Prioritizing Requirements

Prioritizing Requirements

Resources are always limited

Prioritizing requirements can be difficult

Different perspectives are accompanied by differing priorities

Methods for arriving at an objective consensus can help in prioritization

Prioritizing Requirements

Prioritization Methods

- MoSCoW
- Multivoting
- Time-Boxing
- Weighted Ranking

Prioritizing Requirements

MOSCOW

Prioritizing Requirements

Must haves

Fundamental success factors

Should haves

Valuable, but not mission critical

Could haves

Easily left out with minimal impact

Won’t haves

Definitely not being addressed, for now

Prioritizing Requirements

MULTIVOTING

Participants may award votes to each item, based on an established set of rules

Different voting ranks may be used to create a points-based system

Cumulative results of voting lead to a prioritized set of requirements

Prioritizing Requirements

Develop new missions

Develop new items

Develop new social features

Develop new cooperative mode

Develop seasonal bonuses

Prioritizing Requirements

Develop new missions

Develop new items

Develop new social features

Develop new cooperative mode

Develop seasonal bonuses

- High Priority
- Moderate Priority
- Low Priority

Prioritizing Requirements

Time-Boxing

Considers the duration of requirements, and how limited time available can be best maximized

Similar method may be used for budgetary consideration

Prioritizing Requirements

Project Timeline

- Cooperative Mode
- Missions
- New Items
- Social Features
- Seasonal Bonuses

Prioritizing Requirements

Project Timeline

Missions	Social Features	Cooperative Mode	Seasonal Bonuses

New Items

Prioritizing Requirements

Weighted Ranking

Ranking factors and weights should be predetermined, to ensure results are as objective as possible

Score thresholds often used to gauge whether a requirement is important enough to merit commitment

Prioritizing Requirements

	Cost-Effectiveness 35% Weight	Technical Feasibility 40% Weight	Operational Feasibility 25% Weight	Total Score	Final Rank
New Missions	(3*.35) = **1.05**	(5*.40) = **2.00**	(3*.25) = **1.05**	4.10	1
New Items	(5*.35) = **1.75**	(2*.40) = **0.80**	(5*.25) = **1.05**	3.60	2
New Social Features	(1*.35) = **0.35**	(3*.40) = **1.20**	(2*.25) = **1.05**	2.60	5
Cooperative Mode	(2*.35) = **0.70**	(4*.40) = **1.60**	(1*.25) = **1.05**	3.35	3
Seasonal Bonuses	(4*.35) = **1.40**	(1*.40) = **0.40**	(3*.25) = **1.05**	2.85	4

Takeaways

Importance of Documentation

Serves as a baseline for the project

Fundamental point of reference for those signing off on – and those building – the solution

Pivotal for project management

Takeaways

Documenting Solutions

Solution requirements may be functional or nonfunctional in nature

Product requirements describe the end result that meets business needs

Project requirements describe conditions related to how the product is created

Takeaways

Categorizing Requirements

Requirements may be filtered by scope, function, priority or testability to ensure they are valid, align with business needs, and aren't duplicative or poorly written

Takeaways

Documenting Assumptions & Constraints

Assumptions may be made when incomplete information is available, when projecting the future, or extrapolating the present

Constraints may apply to either the product or project, and may be geographical, political, cultural, technical, or legal in nature

Takeaways

Writing Effective Requirements

Requirements must exhibit clarity, precision, consistency, correctness and completeness

Requirements should be measurable, feasible, traceable, and testable

Well-written requirements include a condition, subject, imperative, action and object

Takeaways

Prioritizing Requirements

Resources are always limited, and priorities are rarely clear and unified

MoSCoW, multivoting, time-boxing, and weighted ranking are methods of prioritization

What's Next

Validating, Verifying & Approving Requirements