Vision document

A vision document defines the high-level scope and purpose of a program, product, or project. A clear statement of the problem, proposed solution, and the high-level features of a product helps establish expectations and reduce risks. This topic provides an outline of potential content for a vision document.

See Developing a vision for an explanation of how the product owner or business analyst works with stakeholders to develop a vision document. That topic, which is part of the Collaborative Lifecycle Management scenario guidance, describes the vision-development process. This topic outlines typical content for the document. You can copy this outline, paste it into a new document, and use it as the basis for your vision document. Use those portions of this outline that are relevant for your project.

When a team uses IBM® Rational® Requirements Composer, the vision document can be composed of numerous subdocuments, with each one embedded within the vision document. By taking this approach, the team can work on sections more independently. Team members can set attributes, including status, on each section and create trace links between requirements and the individual documents.

To review the steps for creating a document, see Creating rich-text requirements documents and artifacts.

The vision-document outline

1. Introduction

This introduction provides an overview of the entire vision document. It includes the purpose, scope, definitions, acronyms, abbreviations, references, and an overview of the full document.

1.1 **Purpose:** State the purpose of this vision document.

1.2 **Scope:** Briefly describe the scope of this vision document, including which programs, projects, applications, and business processes the document is associated with. Include anything else that this document affects or influences.

1.3 **Definitions, acronyms and abbreviations:** Define all terms, acronyms, and abbreviations that are required to interpret the vision correctly. This information might be provided by reference to the project glossary, which can be developed online in the Rational Requirements Composer repository.

1.4 **References:** List all documents that the vision document refers to. Identify each document by title, report number (if applicable), date, and publishing organization. Specify the sources from which readers can obtain the references; the sources are ideally available in Rational Requirements Composer or in other online repositories. This information might be provided by reference to an appendix or to another document.

1.5 **Overview:** Describe the vision-document contents and explain how the document is organized.

2. Positioning

2.1 **Business opportunity:** Briefly describe the business opportunity being addressed by this project.

2.2 **Problem statement:** Summarize the problem that this project solves. Use the following statements as a model, providing project details to replace the parenthetical elements:

The problem of (describe the problem) affects (the stakeholders affected by the problem). The impact of the problem is (what is the impact of the problem). A successful solution would include (list some key benefits of a successful solution).

2.3 **Product position statement:** Provide an overall statement that summarizes at the highest level the unique position the product intends to fill in the marketplace. Use the following statements as a model, providing project details to replace the parenthetical elements:

For the (target customer), who (statement of the need or opportunity). The (product name) is a (product category) that (statement of key benefit, that is, the compelling reason to buy). Unlike (primary competitive alternative), our product (statement of primary differentiation).

A product position statement communicates the intent of the application and the importance of the project to all concerned stakeholders.

3. Stakeholder and user descriptions

To provide products and services that meet stakeholders' and users' needs, you must identify and involve all stakeholders

as part of the requirements-definition process. You must also identify the system users and ensure that the stakeholder community represents them adequately.
This section provides a profile of the stakeholders and users who are involved in the project. This section also identifies the key problems that stakeholders and users consider that the proposed solution must address. This section does not describe specific requests or requirements; a separate stakeholder requests artifact captures these items. The key-problem description provides the background and justification for requirements.
3.1 **Market demographics:** Summarize the key market demographics that motivate your product decisions. Describe and position target market segments. Estimate the market size and growth by using the number of potential users. Alternatively, estimate the amount of money that your customers spend trying to meet the needs that your product or enhancement would fulfill. Review major industry trends and technologies. Answer these strategic questions:
- What is the reputation of your organization in these markets?
- What would you like the reputation to be?
- How does this product or service support your goals?

3.2 **Stakeholder summary:** List all the identified stakeholders. For each stakeholder type, provide this information:
- Name: Name the stakeholder type.
- Represents: Briefly describe which individuals, teams, or organizations this stakeholder type represents.
- Role: Briefly describe the role this stakeholder type plays in the development effort.

3.3 **User summary:** List all the identified user types. For each user type, provide this information:
- Name: Name the user type
- Description: Briefly describe the relationship of this type of user to the system under development.
- Stakeholder: List which stakeholder type represents this user type.

3.4 **User environment:** Detail the working environment of the target user. Here are some suggestions:
- How many people are involved in completing the task? Is this changing?
- How long is a task cycle? How much time do users spend in each activity? Is this changing?
- What unique environmental constraints affect the project? For example, do users require mobile devices, work outdoors, or work during flights?
- Which system platforms are in use today? Are there future platforms planned?
- What other applications are in use? Does your application need to integrate with them?

  In this section, you might include extracts from the business model to outline the task and workers who are involved.

3.5 **Stakeholder profiles:** Describe each stakeholder in the project by completing the following table for each stakeholder. Remember: Stakeholder types can be users, strategy departments, legal or compliance departments, technical developers, operations teams, and others. A thorough profile covers the following topics for each stakeholder type:
- Representative: State who represents the stakeholder to the project (This information is optional if it is documented elsewhere.) Enter the representatives' names.
- Description: Briefly describe the stakeholder type.
- Type: Qualify the expertise of the stakeholder, such as "guru," "business expert," , or "casual user." This designation can suggest technical background and degree of sophistication.
- Responsibilities: List the key responsibilities of the stakeholder on the system under development; list their interests as a stakeholder.
- Success criteria: State how the stakeholder defines success. How is the stakeholder rewarded?
- Involvement - Describe how the stakeholder is involved in the project. Where possible, relate the involvement to the process roles; for example, a stakeholder might be a requirements reviewer.

- Deliverables: Identify additional deliverables that the stakeholder requires. These items might be project deliverables or output from the system under development.
- Comments or issues: State problems that interfere with success and any other relevant information.

3.6 **User profiles:** Describe each user of the system here by completing the following table for each user type. Remember user types can be experts and novices; for example, an expert might need a sophisticated, flexible tool with cross-platform support, while a novice might need a tool that is easy to use. A thorough profile covers these topics for each type of user:
- Representative: State who represents the user to the project. (This information is optional if it is documented elsewhere.) This representative often refers to the stakeholder who represents a set of users; for example, Stakeholder: Stakeholder1.
- Description: Briefly describe the user type.
- Type: Qualify the expertise of the user, such as "guru" or "casual user." This designation can suggest technical background and degree of sophistication.
- Responsibilities: List the key user responsibilities with respect to the system; for example, state who captures customer details, produces reports, and coordinates work, and so on.
- Success criteria: State how the user defines success. How is the user rewarded?
- Involvement: Describe how the user is involved in the project. Where possible, relate the involvement to process roles; for example, a stakeholder might be a requirements reviewer.
- Deliverables: Identify the deliverables that the user produces and for whom.
- Comments or issues: State problems that interfere with success and any other relevant information. Describe trends that make the user's job easier or harder.

3.7 **Key stakeholder or user needs:** List the key problems with existing solutions as the stakeholder perceives them. Clarify these issues for each problem:
- What are the reasons for this problem?
- How is the problem solved now?
- What solutions does the stakeholder want?

You must understand the relative importance that the stakeholder places on solving each problem. Ranking and cumulative voting techniques help indicate the problems that must be solved versus issues that stakeholders would like to be addressed. Use this table to capture the stakeholder needs. *Table 1. Stakeholder needs*

Need	Priority	Concerns	Current solution	Proposed solution

3.8 **Alternatives and competition:** Identify alternatives that the stakeholder perceives as available. These alternatives can include buying a competitor's product, building a homegrown solution, or maintaining the status quo. List any known and available competitive choices. Include the major strengths and weaknesses of each competitor as the stakeholder perceives them.

4. Product overview

This section provides a high-level view of the product capabilities, interfaces to other applications, and systems configurations. This section typically consists of three subsections:
- Product perspective
- Product functions

- Assumptions and dependencies

4.1 **Product perspective:** Put the product in perspective with regards to other related products and the user's environment. If the product is independent and totally self-contained, state it here. If the product is a component of a larger system, relate how these systems interact and identify the relevant interfaces between the systems. One way to display the major components of the larger system, interconnections, and external interfaces is to use a business process or use-case diagram.
4.2 **Summary of capabilities:** Summarize the major benefits and features that the product will provide. For example, a customer support system might use this part to address problem documentation, routing, and status reporting without elaborating on detail that these functions require. Organize the functions so that the list is understandable to the customer or to anyone else who reads the document for the first time. A simple table that lists the key benefits and their supporting features might suffice, as in the following example. *Table 2. Benefits and features example*

Customer benefit	Supporting features
New support staff can quickly learn how to use the product.	A knowledge base assists support personnel in quickly identifying known fixes and workarounds.
Customer satisfaction is improved because nothing falls through the cracks.	Problems are uniquely itemized, classified, and tracked throughout the resolution process. Automatic notification occurs for any aging issues.
Management can identify problem areas and gauge staff workload.	Trend and distribution reports enable a high-level review of problem status.
Distributed support teams can work together to solve problems.	With a replication server, current database information can be shared throughout the enterprise.
Customers can help themselves, lowering support costs and improving response time.	A knowledge base can be made available over the Internet. The knowledge base includes hypertext search capabilities and a graphical query engine.

4.3 **Assumptions and dependencies:** List each of factor that affects the features that the vision document includes. List assumptions that, if changed, will alter the vision document. For example, an assumption might state that a specific operating system will be available for the designated hardware for the software product. If the operating system is not available, the vision document will require change.
4.4 **Cost and pricing:** Record relevant cost and pricing impacts and constraints. For example, distribution costs (the number of CDs and CD mastering) or other cost-of-goods-sold constraints (manuals and packaging) might be material or irrelevant to project success, depending on the nature of the application.
4.5 **Licensing and installation:** Licensing and installation issues can also directly affect the development effort. For example, the need to support serializing, password security, or network licensing will create additional system requirements that must be considered in the development effort. Installation requirements might also affect coding, or create the need for separate installation software.

5. Product features

List and briefly describe the product features. Features are the high-level capabilities of the system that are required to deliver benefits to the users. Each feature is a requested service that typically requires a series of inputs to achieve a satisfactory result. For example, a feature of a problem-tracking system might be the ability to provide trending reports. As the use-case model takes shape, update the description to refer to the use cases.
Because the vision document is reviewed by a wide variety of involved personnel, keep the level of detail general enough for everyone to understand. However, offer sufficient detail to provide the team with the information it needs to create a

use-case model or other design documents.
To manage application complexity, for a new system or an incremental change, list capabilities at such a high level that you include approximately 25-99 features. These features provide the basis for product definition, scope management, and project management. Each feature will be expanded into greater detail in the use-case model.
Throughout this section, make each feature relevant to users, operators, or other external systems. Include a description of functions and usability issues that must be addressed. The following guidelines apply:
- Avoid design. Keep feature descriptions at a general level. Focus on required capabilities and why (not how) they should be implemented.
- Designate all features as requirements of a specific feature type for easy reference and tracking.

5.1 **Feature 1**
5.2 **Feature 2**

6. Constraints

Note any design constraints, external constraints, such as operational or regulatory requirements, or other dependencies.

7. Quality ranges

Define the quality ranges for performance, robustness, fault tolerance, usability, and similar characteristics that the feature set does not describe.

8. Precedence and priority

Define the priority of the different system features.

9. Other product requirements

At a high level, list applicable standards, hardware or platform requirements, performance requirements, and environmental requirements.
9.1 **Applicable standards:** List all standards that the product must comply with. The list can include these standards:
- Legal and regulatory standards (FDA, UCC)
- Communications standards (TCP/IP, ISDN)
- Platform compliance standards (Windows, UNIX, and so on)
- Quality and safety standards (UL, ISO, CMM)

9.2 **System requirements:** Define the system requirements for the application. These can include the supported host operating systems and network platforms, configurations, memory, peripheral devices, and companion software.
9.3 **Performance requirements:** Detail performance requirements. Performance issues can include such items as user-load factors, bandwidth or communication capacity, throughput, accuracy, reliability, or response times under various load conditions.
9.4 **Environmental requirements:** Detail environmental requirements as needed. For hardware-based systems, environmental issues can include temperature, shock, humidity, and radiation. For software applications, environmental factors can include use conditions, user environment, resource availability, maintenance issues, error handling, and recovery.

10. Documentation Requirements

This section describes the documentation that you must develop to support successful application deployment.
10.1 **Release notes, read me file:** Release notes or an abbreviated read me file can include a "What's new" section, a discussion of compatibility issues with earlier releases, and installation and upgrade alerts. The document can also contain or link to fixes in the release and any known problems and workarounds.
10.2 **Online help:** Many applications provide an online help system to assist the user. The nature of these systems is unique to application development as they combine aspects of programming (searchable information centers and web-like

navigation) with aspects of technical writing (organization and presentation). Many teams find that developing an online help system is a project within a project that benefits from scope management and planning at the project outset.

10.3 **Installation guides:** A document that includes installation, configuration, and upgrade instructions is part of offering a full solution.

10.4 **Labeling and packaging:** A consistent look and feel begins with product packaging and applies to installation menus, splash screens, help systems, GUI dialog boxes, and so on. This section defines the needs and types of labeling to be incorporated in the code. Examples include copyright and patent notices, corporate logos, standardized icons, and other graphic elements.

11. Appendix 1 - Feature attributes

Give features attributes that can be used to evaluate, track, prioritize and manage the product items that are proposed for implementation. Outline all requirement types and attributes in a separate requirements management plan. However, you might want to list and briefly describe the attributes for features that have been chosen. The following subsections represent a set of suggested feature attributes.

11.1 **Status:** Teams set feature status after negotiation and review by the project management team. Status tracks progress throughout the life of the project. The following table provides an example of typical status-attribute values. *Table 3. Status value examples*

Status	Description
Proposed	Describes features that are under discussion but have not been reviewed and accepted by the "official channel." The official channel might be a working group that consists of representatives from the project team, product management, and user or customer community.
Approved	Capabilities that are deemed useful and feasible and have been approved for implementation by the official channel.
Incorporated	Features that have been incorporated into the product baseline.

11.2 **Benefit:** The marketing group, the product manager, or the business analyst sets the feature benefits. All requirements are not created equal. Ranking requirements by their relative benefit to the user opens a dialog with customers, analysts, and members of the development team. Use benefits in managing project scope and determining development priority. The following table provides an example of typical benefit or priority attribute values. *Table 4. Benefit priority examples*

Priority	Description
Critical	Essential features. Failure to implement a critical feature means that the system will not meet customer needs. All critical features must be implemented in the release or the schedule will slip.
Important	Features important to the effectiveness and efficiency of the system for most applications. The functions cannot be easily provided in some other way. Omitting an important feature might affect customer or user satisfaction, or even revenue. However, the release will not be delayed because an important feature is not included.
Useful	Features that are useful in less typical applications, are used less frequently, or that can be met with reasonably efficient workarounds. No significant revenue or customer satisfaction impact can be expected if such an item is not included in a release.

11.3 **Effort:** The development team estimates the effort that is required to implement features. Some features require more time and resources than others. Estimating the time, required code, or functions, helps gauge complexity and set expectations of what can be accomplished in a given time frame. Use the estimate in managing scope and determining development priority.

11.4 **Risk:** The development team establishes risk levels, based on the probability that the project will experience undesirable events, such as cost overruns, schedule delays, or even cancellation. Most project managers find categorizing risks as high, medium, and low is sufficient, although finer gradations are possible. Risk can often be assessed indirectly by measuring the uncertainty (range) of the project team's schedule estimate.

11.5 **Stability:** The analyst and development team establish feature stability based on the probability that the feature will change or the team's understanding of the feature will change. Stability is used to help establish development priorities and determine those items for which additional elicitation is the appropriate next action.

11.6 **Target release:** Teams record the earliest intended product version that will include the feature. You can use this field to allocate features from a vision document into a particular baseline release. When combined with the status field, your team can propose, record, and discuss various features of the release without committing them to development. Only features whose status is set to "incorporated" and whose target release is defined will be implemented. With scope management, the target release version number can be increased, and the item remains in the vision document but is scheduled for a later release.

11.7 **Assigned to:** In many projects, features are assigned to feature teams that are responsible for further elicitation, writing the software requirements, and implementation. The process helps everyone on the project team better understand responsibilities.

11.8 **Reason:** Teams use this text field to track the source of the requested feature. Requirements exist for specific reasons. This field records an explanation or a reference to an explanation. For example, the reference might point to a page and line number of a product requirement specification or point to a minute marker on a customer-interview video.