Volere

Requirements Specification Template

Edition 16—2012

by James Robertson & Suzanne Robertson
principals of the Atlantic Systems Guild

The Volere Requirements Specification Template is intended for use as a basis for your requirements specifications. The template provides sections for each of the requirements types appropriate to today's software systems. You may download the template from the Volere site and adapt it to your requirements gathering process and requirements tool. The template can be used with Requisite, DOORS, Caliber RM, IRqA and other popular tools see http://www.volere.co.uk/tools.htm

Contents

The Volere Requirements Knowledge Model (included with the download of Version 16 of this template) shows the formal structure and cross references between the components in the above table of contents.

Volere

Volere is the result of many years of practice, consulting, and research in requirements engineering and business analysis. We have packaged our experience in the form of a generic requirements process, requirements training, requirements consultancy, requirements audits, a variety of downloadable guides and articles, a requirements knowledge model and this requirements template. We also provide requirements specification-writing services.

The first edition of the Volere Requirements Specification Template was released in 1995. Since then, organizations from all over the world have saved time and money by using the template as the basis for discovering, organizing, and communicating their requirements.

The Volere web site www.volere.co.uk contains articles about the Volere techniques, experiences of Volere users and case studies, requirements tools, and other information useful to requirements practitioners.

The Volere requirements process is described in the book *Mastering the Requirements Process—Second Edition* by Suzanne Robertson and James Robertson, Addison-Wesley, 2006. ISBN 0-321-41949-9

For more about managing requirements see *Requirements Led Project Management* by Suzanne Robertson and James Robertson, Addison-Wesley, 2005. ISBN 0-321-65904-X

Updates to this template and instructions for downloading are available at http://www.volere.co.uk

Public seminars on Volere are run on a regular basis in Europe, the United States, Australia, and New Zealand. For a schedule of courses, refer to www.volere.co.uk.

Requirements Types

For ease of use, we have found it convenient to think of requirements as belonging to a type. There are two reasons for the type: as an aid to finding the requirements, to be able to group the requirements that are relevant to a specific expert specialty.

Functional requirements are the fundamental or essential subject matter of the product. They describe what the product has to do or what processing actions it must take.

Non-functional requirements are the properties that the functions must have, such as performance and usability. Do not be deterred by the unfortunate name for this kind of requirements, they are as important as the functional requirements for the product's success.

Project constraints are restrictions on the product due to the budget or the time available to build the product.

Design constraints impose restrictions on how the product must be designed. For example, it might have to be implemented in the hand-held device being given to major customers, or it might have to use the existing servers and desktop computers, or any other hardware, software, or business practice.

Project drivers are the business-related forces. For example, the purpose of the project is a project driver, as are all of the stakeholders—each for different reasons.

Project issues define the conditions under which the project will be done. Our reason for including them as part of the requirements is to present a coherent picture of all factors that contribute to the success or failure of the project and to illustrate how managers can use requirements as input when managing a project.

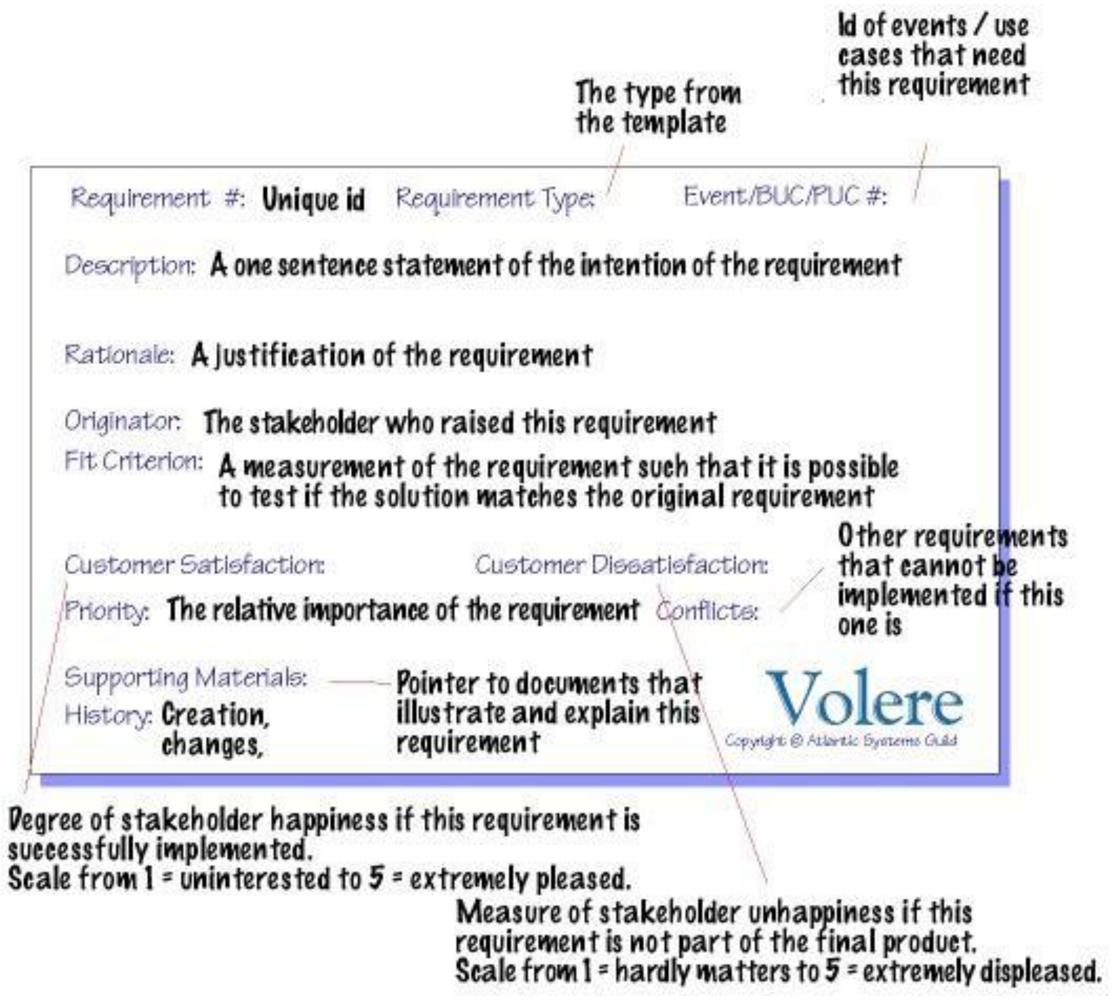
Testing Requirements

The Volere philosophy is to start testing requirements as soon as you start writing them. You make a requirement testable by adding its *fit criterion*. This fit criterion measures the requirement, making it possible to determine whether a given solution fits the requirement. If a fit criterion cannot be found for a requirement, then the requirement is either ambiguous or poorly understood. All requirements can be measured, and all should carry a fit criterion.

Atomic Requirements Shell

The requirements shell is a guide to writing each atomic requirement. The components of the shell (also called a "snow card") are identified below. You might decide to add some additional attributes to provide traceability necessary for your environment. For example: products that implement this requirement, version of the software that implements this requirement, departments who are interested in this requirement, etc. There are others but do not capriciously add attributes unless they really help you: every attribute you add needs to be maintained.

This requirements shell can, and should, be automated. When you download the template you will also find an Excel spreadsheet implementation of the snow card.



Id of events / use cases that need this requirement

The type from the template

Requirement #: Unique id Requirement Type: Event/BUC/PUC #:

Description: A one sentence statement of the intention of the requirement

Rationale: A justification of the requirement

Originator: The stakeholder who raised this requirement

Fit Criterion: A measurement of the requirement such that it is possible to test if the solution matches the original requirement

Customer Satisfaction: Customer Dissatisfaction:

Priority: The relative importance of the requirement Conflicts:

Other requirements that cannot be implemented if this one is

Supporting Materials: Pointer to documents that illustrate and explain this requirement

History: Creation, changes,

Volere

Copyright © Atlantic Systems Guild

Degree of stakeholder happiness if this requirement is successfully implemented. Scale from 1 = uninterested to 5 = extremely pleased.

Measure of stakeholder unhappiness if this requirement is not part of the final product. Scale from 1 = hardly matters to 5 = extremely displeased.

The following discusses and provides examples for each of the sections of the Volere Requirements Specification Template. For each section, the Content, Motivation, Considerations, Examples and Form provide the template user with some guidance for writing each type of requirement. When you download the template you will also find a Template Skeleton that you might find convenient to use as the basis for producing a document.

1. The Purpose of the Project

The first section of the template deals with the fundamental reason your client asked you to build a new product. That is, it describes the business problem the client faces and explains how the product is intended to solve the problem.

1a. The User Business or Background of the Project Effort

Content

The university currently relies on manual verification by gate attendants to manage parking access for students and faculty. This outdated process causes significant congestion and long wait times at entry points, as every vehicle must be stopped and checked. Additionally, the university has installed smart sensors that remain underutilized, meaning there is currently no way to track available spots in real-time. These inefficiencies and the need to modernize campus infrastructure triggered the development of the Automated University Garage Management System (AUGMS).

The work that the users intend to do with the delivered product is to fully automate the parking experience. The system will control vehicle entry using license plate recognition, allowing verified users to enter without stopping. Students and faculty will use the platform to check for empty spots before arriving and to book services like car cleaning. Administrators will use the system to monitor garage occupancy levels and manage user registrations efficiently.

Motivation

The motivation for this project is to eliminate the severe congestion and delays caused by the current manual verification process, which frustrates students and faculty daily. Additionally, the university aims to justify its investment in existing smart sensor infrastructure by finally integrating these underutilized devices into a functional system to modernize campus operations.

Considerations

The business problem is serious because the current manual process causes daily traffic congestion, which frequently makes students and faculty late for their academic duties. This inefficiency disrupts the university's daily schedule and frustrates the community. Additionally, there is a significant business opportunity to utilize the smart sensors that the university has already purchased but left inactive. Solving this problem is necessary to eliminate these avoidable delays and to justify the previous financial investment in the sensor infrastructure.

Form

The project motivation is documented through a short text description summarizing the current inefficiencies. This is supported by an **Activity Diagram** (Deliverable 1, p. 13) that models the proposed vehicle entry and verification workflow.

1b. Goals of the Project

Content

The project aims to deploy a fully automated parking management system that eliminates the need for manual gate checks and optimizes garage usage. The product will autonomously control vehicle access using license plate recognition and provide real-time occupancy data to drivers. This brings a critical advantage to the university by drastically reducing entry wait times, preventing traffic bottlenecks at campus gates, and modernizing infrastructure through the integration of previously unused smart sensors.

Motivation

The primary motivation is to reclaim the time currently lost by students and faculty due to inefficient manual parking checks. If the system fails to prioritize speed and automation, the core problem of daily congestion will remain unsolved, rendering the project a failure regardless of its other features. Furthermore, the university has a financial imperative to operationalize the expensive smart sensors that are currently sitting idle; failing to do so represents a continued waste of resources and an inability to modernize campus infrastructure.

Examples

We want to reduce vehicle   entry time to under 5 seconds by automating the verification process.

To eliminate the daily traffic congestion at university gates by removing the need for manual checks.

To utilize the existing smart sensor infrastructure to provide real-time parking availability data to students and faculty.

.

Measurement

The primary measurement of success is the speed of vehicle access. We will quantify this by testing if the system can process a license plate and grant entry in under 2 seconds, which is significantly faster than the current manual method. We will also measure reliability by tracking system downtime, with a specific goal of maintaining 99% operational uptime during university hours to ensure that automation does not create new delays

Form

To implement a fully automated garage system that solves daily traffic congestion and finally activates the university's dormant smart sensor infrastructure. **Advantage:** The university will eliminate long queues at entry gates and reduce the manual labor required for verification, ensuring students and faculty spend less time parking and more time on academic activities. **Measurement:** Vehicle verification time reduced to < 2 seconds per car; System uptime maintained at ≥ 99%.

2. The Stakeholders

This section describes the stakeholders—the people who have an interest in the product. It is worth your while to spend enough time to accurately determine and describe these people, as the penalty for not knowing who they are can be very high.

2a. The Client

Content

The client for this project is the **University Management**. This executive body is responsible for campus operations and infrastructure and is the primary sponsor authorizing the development of the Automated University Garage Management System to solve the current parking inefficiencies.

Motivation

The University Management is the sole authority for funding and accepting the final system. As the body making the financial investment in the AUGMS project, their satisfaction with the delivered solution is the ultimate criterion for success. Without their explicit approval and alignment with the project goals, the development effort lacks legitimate authorization and direction.

Considerations

Since this system is being developed for internal use within the university rather than as a commercial product for the general public, the "Marketing Department" does not act as the client. Instead, the considerations focus on ensuring that the University Management designates a specific representative with the actual authority to make decisions. This is critical to avoid bureaucratic delays where approval for requirements or changes might otherwise get stuck in administrative processing.

Form

An organization chart is included showing the University Management at the executive top level, with the Security Department (users) and IT Department (technical support) reporting directly to them.

The client is responsible for the following decisions:

- Authorization of the project budget and hardware procurement (sensors and cameras).
- Approval of the functional requirements outlined in Deliverable 1.
- Final sign-off and acceptance of the system for campus deployment.

The project progress is tracked through the following review checkpoints:

- **Checkpoint 1:** Review of System Vision and Requirements (Deliverable 1 - Completed).
- **Checkpoint 2:** Review of User Stories and Use Cases (Deliverable 2 - Nov 16, 2025).
- **Checkpoint 3:** Final Prototype Demonstration and Acceptance (End of Semester).

.

2b. The Customer

Content

For this in-house development project, the customer is the University Management, specifically the Campus Operations Department. As the body responsible for financing campus infrastructure and security upgrades, they act as the purchaser of the system. They are responsible for allocating the budget for the software development and the procurement of the required hardware (cameras and sensors)).

Motivation

The customer must be clearly identified because they control the funding. If the University Management does not see a return on investment—specifically in the form of reduced congestion and utilized hardware—they will not release the funds required to complete the project. Understanding their financial constraints and operational goals is essential to ensure the project remains viable.

Form

The customer is represented by the Head of Campus Operations in the organizational chart. The agreement is formalized through the internal Project Budget Approval Document, which authorizes the spending for the AUGMS development.

2c. Other Stakeholders

Content

Beyond the client and customer, several other groups are critical to the system's success.

- **Students and Faculty (End Users):** These are the daily drivers using the garage. We need to obtain knowledge about their arrival times, parking habits, and service preferences (like car cleaning). They require high involvement during requirements gathering (via interviews/surveys) but have low influence on the final budget.
- **Security Department & Garage Staff:** These are the people physically operating the gates and monitoring the system. We need their knowledge regarding current security protocols, manual override procedures, and shift schedules. They have high influence on the design of the admin dashboard to ensure it is usable.
- **IT Department:** This internal team supports the university's network. We need their knowledge regarding server hosting, network security standards, and database policies. Their influence is high regarding the technical architecture and deployment constraints.
- **External Sensor Providers:** These vendors supply the smart hardware. We need technical documentation and API specifications from them to ensure our software can communicate with their cameras and sensors.

Motivation

We identify these stakeholders because the system must work for the people who use it daily, not just for the management buying it. If the Security Staff finds the dashboard too difficult to use, or if the IT Department refuses to host the software due to security risks, the project will fail regardless of executive approval.

Form

The stakeholders are documented in a Stakeholder Analysis Matrix which maps each role to their responsibility. We also use an Onion Diagram to visualize the relationship between the core development team, the internal university stakeholders (Security/IT), and the external world (Vendors)

2d. The Hands-On Users of the Product

Content

<u>User Category: University Students and Faculty</u>

- **User Role:** Daily end-users who enter and exit the garage. Responsibilities include registering vehicles, checking real-time parking availability via the mobile app, and requesting add-on services like car cleaning.
- **Subject Matter Experience: Journeyman.** They understand the general parking rules and layout of the university but do not know the operational protocols or security limitations.
- **Technological Experience: Master.** Students are digital natives highly proficient with mobile applications; faculty are generally proficient with university systems.
- **Other User Characteristics:**
    - **Attitude toward technology:** High expectations for speed and responsiveness; low tolerance for bugs or complex menus.
    - **Physical Location:** Often using the application outdoors while walking to the car or inside a vehicle; the interface must be usable in motion.
    - **Linguistic Skills:** Bilingual (Arabic and English). The interface must support both languages.

<u>User Category: Security Guards & Garage Staff</u>

- **User Role:** Operational staff stationed at the gates. Responsibilities include monitoring automated entry, performing manual overrides for unrecognized vehicles, and handling physical security incidents.
- **Subject Matter Experience: Master.** They have deep knowledge of peak traffic times, physical bottlenecks, and emergency procedures.
- **Technological Experience: Novice to Journeyman.** They view technology as a tool to assist their physical duties and may not be comfortable with complex troubleshooting.
- **Other User Characteristics:**
    - **Physical Location:** Outdoors at entry gates. Devices must be readable in bright sunlight and resistant to dust and heat.
    - **Attitude toward job:** Focused on security and maintaining flow; they require instant feedback from the system to prevent backing up traffic.

    - **Education:** Varies; the interface must be simple, using clear icons rather than dense text to ensure quick understanding.

User Category: System Administrators

- **User Role:** Back-office management. Responsibilities include approving user registrations, generating utilization reports for the university board, and configuring system parameters (e.g., parking fees or operating hours).
- **Subject Matter Experience: Master.** They define the business rules and access policies for the garage.
- **Technological Experience: Master.** Likely IT professionals or trained administrators capable of handling data analytics and complex dashboards.
- **Other User Characteristics:**
    - **Physical Location:** Indoor office environment, using desktop computers.
    - **Intellectual Abilities:** Capable of analyzing statistical data and charts to make long-term operational decisions.

Motivation

We analyze these user characteristics to define the specific usability requirements for each group. Since AUGMS users operate in vastly different environments—from students rushing to class to guards standing in outdoor heat—a generic interface would fail. Understanding these distinct human factors allows us to design interfaces that fit their specific physical contexts and technical skills, ensuring the system aids their daily tasks rather than hindering them..

Examples

Casual User: Mohamed Ehab, a 3rd-year CIS student who accesses the mobile app daily to find parking before lectures.

Business User: A Faculty Member who uses the system to book car cleaning services during their office hours.

Operational User: A Security Guard stationed at the gate who interacts with the ruggedized tablet to handle manual overrides for unrecognized vehicles.

Administrative User: The IT Manager who accesses the system remotely from the back office to update the authorized vehicle database

.

Form

The user characteristics are documented in the following list, linking each user role to a specific representative persona used during our requirements gathering:

- **Student User:** Represented by **"Yousef,"** a 4th-year CIS student.
    - *Characteristics:* High tech literacy, uses mobile app daily, time-poor.
- **Faculty User:** Represented by **"Dr. Nourhan,"** a junior professor.
    - *Characteristics:* Moderate tech literacy, requires reliable spot reservation, values efficiency.
- **Security Guard:** Represented by **"Captain Mahmoud,"** a veteran gatekeeper.
    - *Characteristics:* Low tech literacy, works outdoors in heat/dust, needs large buttons and simple alerts.
- **System Administrator:** Represented by **"Eng. Hassan,"** the IT specialist.
    - *Characteristics:* Expert tech literacy, works in an office, manages database and reporting configuration.

2e. Personas

Content

We have developed specific personas to represent the key user groups. These fictional characters help the development team visualize who they are building for, ensuring the system meets real human needs rather than abstract requirements.

Persona 1: The Rushing Student

- **Name:** Ahmed "The Sprinter"
- **Age:** 21
- **Job:** Senior CIS Student
- **Family:** Lives with parents in Nasr City.
- **Hobbies:** Gaming, Gym, and Coding Hackathons.
- **Attitude to Technology: High.** He is an early adopter who gets frustrated if an app takes more than 3 seconds to load. He expects everything to be "one-click."
- **Attitude to Money:** Budget-conscious student; hates paying for parking but willing to pay for convenience if it saves him from being late.
- **Likes:** Efficiency, Dark Mode, Automated notifications.
- **Dislikes:** Waiting in lines, manual paperwork, carrying cash.
- **Influence on Product:** The mobile app must have a "Quick View"

widget for parking spots so he can check it while driving (safely) or running to class.

Persona 2: The Veteran Gatekeeper

- **Name:** Captain Mahmoud
- **Age:** 52
- **Job:** Senior Security Guard
- **Family:** Married with 3 children.
- **Hobbies:** Watching football matches at the local cafe.
- **Attitude to Technology: Low/Skeptical.** He prefers his radio and clipboard because they "never crash." He is worried that the new system will be complicated or make him look incompetent.
- **Attitude to Money:** Conservative; values job security.
- **Likes:** Simple instructions, large buttons, reliability.
- **Dislikes:** Small text, complicated logins, devices that stop working in the heat.
- **Influence on Product:** The guard interface must be rugged, high-contrast (for sunlight), and require minimal tapping to perform a manual override.

Persona 3: The Efficient Professor

- **Name:** Dr. Sarah
- **Age:** 38
- **Job:** Junior Professor
- **Family:** Married, mother of two toddlers.
- **Hobbies:** Reading, Yoga.
- **Attitude to Technology: Moderate/Utilitarian.** She uses technology as a tool to organize her chaotic schedule. If it works, she loves it; if it breaks, she abandons it immediately.
- **Influence on Product:** She needs a reliable reservation system or guaranteed availability status so she can drop off her kids and arrive exactly on time for her lecture without stress.

.

Motivation

We utilize these three personas to prevent the development team from designing for a generic "average user" who does not actually exist. By visualizing specific characters like "Captain Mahmoud" operating the system in the outdoor heat, or "Ahmed" rushing to class, we ensure the requirements address real human constraints—such as screen glare, physical dexterity, and time pressure—rather than just satisfying abstract technical functions. This technique keeps the team focused on satisfying specific people with distinct needs.

Form

We have created **Persona Profile Cards** for Ahmed, Captain Mahmoud, and Dr. Sarah. These are formatted as single-page documents shared on our Jira dashboard and printed for design meetings.

Each profile contains:

- **Visuals:** A representative photograph (e.g., a student with a backpack, a guard in uniform) to make the user "real" to the developers.
- **Narrative:** A "Day in the Life" story describing their typical schedule and how they currently interact with the garage.
- **Key Drivers:** A bulleted list of their specific "Goals" (what they want) and "Frustrations" (what they hate about the current system).

These cards serve as the constant reference point for all design decisions

.

2f. Priorities Assigned to Users

Content

We have classified the users based on their impact on the system's operational success and security.

- **Key Users: Security Guards and Garage Staff.** These users are critical because they control the physical entry points. If the system fails them or is too difficult to operate during a rush, the entire purpose of "reducing congestion" fails immediately. Their requirements for reliability, speed, and manual override capabilities are given the highest importance.
- **Key Users: System Administrators.** They are critical for the long-term maintenance and data integrity of the system. Their ability to manage the database and configure rules ensures the system remains functional.
- **Secondary Users: Students and Faculty.** While they are the largest group and the primary beneficiaries of the faster service, their role is passive (the system scans them). In a conflict between a Student's desire for convenience (e.g., "let me in even if my plate is dirty") and a Security Guard's need for strict verification, the Security Guard's requirement

takes precedence.

- **Unimportant Users: Unregistered Visitors / Unauthorized Drivers.** This category includes people trying to use the garage without registration. The system is designed to exclude them, so their "requirements" (e.g., guest access) are not currently considered.

Motivation

Stating these priorities is essential to guide design trade-offs. We prioritize the Security Guards' requirements because their ability to quickly manage exceptions at the gate is the critical factor in preventing congestion. If we were to favor the convenience of the Secondary Users (Students) over the operational control required by the Key Users, we risk creating a system that is user-friendly but operationally fragile, leading to the very traffic jams the project aims to eliminate.

Form

The user importance ratings have been integrated into the User Characteristics Spreadsheet. This document is classified as internal confidential information for the core project team only, to prevent potential dissatisfaction among the "Secondary" user groups (Students and Faculty) who might feel undervalued despite being the largest population.

2g. User Participation

Content

We have defined the specific contributions required from each user group to ensure the requirements are accurate and feasible.

- **Students and Faculty:** We require them to participate in online questionnaires and brief interviews to provide data on their arrival times and service preferences (e.g., how much they value car cleaning). This input is vital for the mobile app design.
  - *Minimum Time:* 30 minutes per representative.
- **Security Guards:** We require permission to "shadow" them during peak morning shifts to observe the physical bottlenecks at the gates. They must also participate in usability testing for the guard dashboard to ensure it is usable in outdoor conditions.
  - *Minimum Time:* 4 hours (2 hours observation + 2 hours interview/testing).

- **System Administrators:** We require their participation in technical review sessions to define data security protocols and server hosting constraints. They must validate that our proposed architecture fits within the university's existing IT infrastructure.
    - *Minimum Time:* 5–8 hours spread across the requirement and design phases

.

Motivation

Many projects fail because users prioritize their daily jobs over providing requirements. For AUGMS, this is a significant risk with the Security Guards, who are constantly busy managing active traffic. If we do not obtain a formal agreement for their participation now, they will naturally prioritize their gate duties and be unavailable to validate the outdoor interface. Explicitly allocating this time ensures that their critical feedback on usability is prioritized over their routine duties during the design phase, preventing the delivery of a system that is unusable in real-world conditions.

Form

Include the estimated user participation time, together with the type of knowledge you expect that user to provide, on your user characteristics spreadsheet, see 2d.

2h. Maintenance Users and Service Technicians

Content

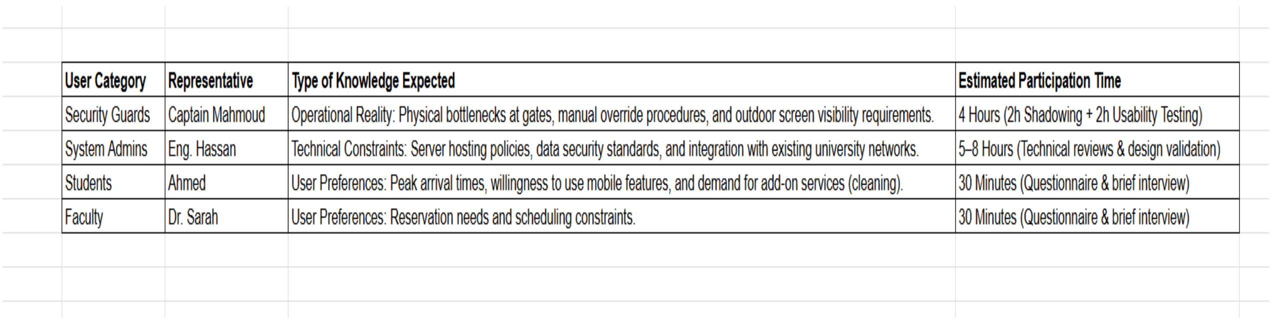
Maintenance users are a special type of hands-on users who have requirements that are specific to maintaining and changing the product.

Motivation

Many of these requirements will be discovered by considering the various types of maintenance requirements detailed in section 14. However, if we define the characteristics of the people who maintain the product, it will help to trigger requirements that might otherwise be missed.

Form

Include the maintenance users, on your user characteristics spreadsheet, see 2d.



User Category	Representative	Type of Knowledge Expected	Estimated Participation Time
Security Guards	Captain Mahmoud	Operational Reality: Physical bottlenecks at gates, manual override procedures, and outdoor screen visibility requirements.	4 Hours (2h Shadowing + 2h Usability Testing)
System Admins	Eng. Hassan	Technical Constraints: Server hosting policies, data security standards, and integration with existing university networks.	5–8 Hours (Technical reviews & design validation)
Students	Ahmed	User Preferences: Peak arrival times, willingness to use mobile features, and demand for add-on services (cleaning).	30 Minutes (Questionnaire & brief interview)
Faculty	Dr. Sarah	User Preferences: Reservation needs and scheduling constraints.	30 Minutes (Questionnaire & brief interview)

3. Mandated Constraints

This section describes constraints on the eventual design of the product. Constraints are global—they are factors that apply to the entire product. The product must be built within the stated constraints. Often you know about the constraints, or they are mandated before the project gets under way. They are probably determined by management and are worth considering carefully—they restrict what you can do and so shape the product. Constraints, like other types of requirements have a description, rationale, and fit criterion, and generally are written in the same format as functional and non-functional requirements.

3a. Solution Constraints

Content

The solution must adhere to the following technological constraints mandated by the university environment:

- **Hardware Integration:** The system must interface with the **existing smart sensors and IP cameras** that are currently installed in the garage.
    - *Reason:* The university has already invested capital in this hardware. The software must utilize this specific equipment to justify the sunk cost and avoid new procurement expenses.
- **Platform Requirement:** The end-user interface for students and faculty must be a **Mobile Application (iOS/Android)**.
    - *Reason:* Users need to check parking availability and receive notifications while in transit or walking to the garage, which is not feasible with a web-only interface.
- **Authentication Source:** The system must integrate with the university's **existing Student/Faculty Database** (e.g., via LDAP or API).
    - *Reason:* To ensure that only currently enrolled students and active staff can register vehicles, preventing unauthorized access by former students.
- **Operating Environment:** The gate control software must run on **ruggedized hardware** capable of withstanding outdoor heat and dust.
    - *Reason:* The physical location of the garage entry is exposed to the elements.

.

Motivation

We must identify these constraints immediately because the university will not accept a solution that requires scrapping their existing hardware or operating outside their security network. If we ignore these mandated technologies—for instance, by building a system that doesn't work with the installed cameras—the final product will be rejected regardless of its quality, resulting in a total project failure.

Examples

Example 1: Existing Hardware Integration

- **Description:** The product shall integrate with the currently installed smart sensors and IP cameras at the garage entry/exit points.
- **Rationale:** The university has already invested significant capital in this hardware and will not approve a budget for replacing functional equipment.
- **Fit Criterion:** The software must successfully receive data streams and identify license plates using the specific API/SDK of the existing sensor models (e.g., Hikvision/Dahua) without requiring hardware replacement.

Example 2: Mobile Platform Requirement

- **Description:** The user interface for students and faculty shall be a native mobile application supported on both iOS and Android platforms.
- **Rationale:** Users need to check parking availability and receive notifications while in transit; a desktop or web-only interface is not usable for drivers.
- **Fit Criterion:** The application must be available for download on App Stores and successfully run on devices with iOS 15+ and Android 12+, with a "Pass" grade on standard usability tests for mobile responsiveness.

Example 3: Database Authentication

- **Description:** The product shall authenticate all user registrations against the existing University Student/Faculty Database (LDAP/Active Directory).
- **Rationale:** To ensure that only currently enrolled students and employed staff can access the garage, preventing security risks from former students or unauthorized public users.
- **Fit Criterion:** A user login attempt must trigger a query to the university's central server; if the ID is flagged as "Inactive" or "Suspended" in the central database, the AUGMS system must deny registration.

Example 4: Environmental Durability

- **Description:** The gate control hardware interfaces must be ruggedized and suitable for outdoor use.
- **Rationale:** The equipment is permanently stationed at the garage entry, exposed to high temperatures, dust, and direct sunlight.
- **Fit Criterion:** The hardware units must have an IP65 rating (dust/water resistance) and the display screens must be readable in 10,000 lux (direct sunlight) conditions.

Considerations

We must be careful to distinguish between *true* constraints (unavoidable limitations) and *preferences* (optional technology). For AUGMS, the use of the **existing smart sensors** is a true constraint because the university has a strict "no new hardware purchase" policy for the gates. However, the specific database (e.g., MySQL vs. PostgreSQL) is *not* a mandated constraint; while the IT team might prefer one, the solution would still be acceptable if it used another, as long as it integrates with the university network. We have therefore only listed constraints that are absolutely non-negotiable by the university administration..

Form

Include the constraint requirements as a specific type of atomic requirement in your requirements spreadsheet or database. For attributes of an atomic requirement see the Atomic Requirements Shell example at the start of this template. Also refer to article on atomic requirements at http://www.volere.co.uk

Another form for constraints can be diagram/s of the systems architecture for the new/changed product – see 3b & 3c.

3b. Implementation Environment of the Current System

Content

This describes the technological and physical environment in which the product is to be installed. It includes automated, mechanical, organizational, and other devices, along with the nonhuman adjacent systems.

Motivation

To describe the technological environment into which the product must fit. The environment places design constraints on the product. This part of the specification provides enough information about the environment for the designers to make the product successfully interact with its surrounding technology.

The operational requirements are derived from this description.

Examples

Examples can be shown as a diagram, with some kind of icon to represent each separate device or person (processor). Add interfaces between the processors, and annotate them with their form and content.

Considerations

All component parts of the current system, regardless of their type, should be included in the description of the implementation environment.

If the product is to affect, or be important to, the current organization, then include an organization chart.

Form

A diagram that represents each hardware and software component/subcomponent/device/building block that will be used to implement the product. The particular diagrams that you use depend on your organization and your projects' ways of working, the important issue is that the implementation environment is unambiguously understandable by the people who have to make decisions about how the functional and non-functional requirements will be implemented. Types of UML diagrams commonly used are: Class, Component, Component Structure, Deployment, Package diagrams. There are many other home-grown diagrams.

3c. Partner or Collaborative Applications

Content

This describes applications that are not part of the product but with which the product will collaborate. For AUGMS, these include the university's existing campus ID authentication system (for user login and verification) and the external Hikvision LPR API for camera integration. AUGMS will interface with these via RESTful APIs for real-time data exchange, such as validating user IDs during registration or fetching live camera feeds.

Motivation

To provide information about design constraints caused by using partner applications. By describing or modelling these partner applications, you discover and highlight potential problems of integration.

Examples

The campus ID system will provide user authentication tokens to AUGMS upon login. The Hikvision API will send license plate data to AUGMS for entry validation.

Considerations

Examine the work context model to determine whether any of the adjacent systems should be treated as partner applications. It might also be necessary to examine some of the details of the work to discover relevant partner applications. For AUGMS, integration with the campus ID system is critical to avoid duplicating user data, but potential mismatches in API versions could cause delays.

Form

A diagram or table that identifies all the interfaces between the product to be built and other adjacent systems. Bear in mind that the adjacent systems might be software, human or hardware. Some adjacent systems are within your organization and hence potentially more easily understood and perhaps influenced. Other adjacent systems are outside your organization and might be difficult if not impossible to influence. A product scope diagram (see 8a for an example) is often used to define interfaces with partner or collaborative applications. For AUGMS, refer to the UML System Sequence Diagrams in Deliverable #3 (Section III.3.2) showing interactions with sensors and cameras.

3d. Off-the-Shelf Software

Content

This describes commercial, open source, or any other off-the-shelf software (OTS) that must be used to implement some of the requirements for the product. For AUGMS, we will use the open-source OpenALPR library for license plate recognition processing and MySQL as the database backend. These OTS components will handle image processing and data storage, respectively.

Motivation

To identify and describe existing commercial, free, open source, or other products to be incorporated into the eventual product. The characteristics, behaviour, and interfaces of the package are design constraints.

Considerations

When gathering requirements, you may discover requirements that conflict with the behaviour and characteristics of the OTS software. Keep in mind that the use of OTS software was mandated before the full extent of the requirements became known. In light of your discoveries, you must consider whether the OTS product is a viable choice. If the use of the OTS software is not negotiable, then the conflicting requirements must be discarded. Note that your strategy for discovering requirements is affected by the decision to use OTS software. In this situation you investigate the work context in parallel with making comparisons with the capabilities of the OTS product. Depending on the comprehensibility of the OTS software, you might be able to discover the matches or mismatches without having to write each of the business requirements in atomic detail. The mismatches are the requirements that you will need to specify so that you can decide whether to satisfy them by either modifying the OTS software or satisfying the requirement in another way or modifying the business requirements. Given the spate of lawsuits in the software arena, you should consider whether any legal implications might arise from your use of OTS. You can cover this in section 17. Legal Requirements.

Examples

OpenALPR will process camera images to extract license plates with ≥95% accuracy in daylight. MySQL will store all Vehicle and LogEntry data as per the domain model.

Form:

Models or written documentation that specifies the functional and non-functional requirements that can be implemented using this OTS software product. If the OTS product has a well structured requirements specification and systems architecture model then that provides you with the basis for identifying which of your requirements can be satisfied by the product. If the product's documentation is not traceable and well organized then you will need to do more detailed work on your own requirements until you find a level at which you can map your requirements to the OTS product. Another form is a person or people who are experts in the OTS product and can answer your questions without you having to puzzle through cryptic or marketing-oriented documents. For AUGMS, refer to the CRUD matrix in Deliverable #3 (Section II.2.1) to map OTS usage to classes like Vehicle and ParkingSpot.

3e. Anticipated Workplace Environment

Content

This describes the workplace in which the users are to work and use the product. It should describe any features of the workplace that could have an effect on the design of the product, and the social and cultural aspects of the workplace. For AUGMS, the primary workplaces are: outdoor entry/exit gates (dusty, bright sunlight, high traffic noise) for security guards using rugged tablets; indoor offices for administrators using desktops; and mobile/on-the-go for students/faculty via smartphones (potentially while driving or walking).

Motivation

To identify characteristics of the workplace so that the product is designed to compensate for any difficulties.

Examples

The gate area is exposed to weather, so devices must be IP65-rated. Mobile app must have high-contrast UI for sunlight readability.

Considerations

The physical work environment constrains the way that work is done. The product should overcome whatever difficulties exist; however, you might consider a redesign of the workplace as an alternative to having the product compensate for it.

Form

Written description of the workplace; rich pictures showing all the components in the workplace; photographs of the workplace; videos of the workplace. For AUGMS, refer to Activity Diagrams in Deliverable #3 (Section III.3.1) with swimlanes showing user environments**.**

3f. Schedule Constraints

Content

Any known deadlines, or windows of opportunity, should be stated here. For AUGMS, the system must be fully deployed by 20 December 2025 (end of Fall semester) to avoid disrupting the Spring semester start. Pilot testing must begin by 1 December 2025.

Motivation

To identify critical times and dates that have an effect on product requirements. If the deadline is short, then the requirements must be kept to whatever can be built within the time allowed.

Examples

To meet scheduled software releases. There may be other parts of the business or other software products that are dependent on this product. Windows of marketing opportunity. Scheduled changes to the business that will use your product. For example, the organization may be starting up a new factory and your product is needed before production can commence.

Considerations

State deadline limitations by giving the date and describing why it is critical. Also, identify prior dates where parts of your product need to be available for testing. You should also ask questions about the impact of not meeting the deadline: What happens if we don't build the product by the end of the calendar year? What is the financial impact of not having the product by the beginning of the Christmas buying season? What parts of the product are most critical for the Christmas buying season?

Form

A written statement giving: The date of the deadline, The reason for the deadline, The effect of not meeting the deadline. For AUGMS, a Gantt chart from project management tools, cross-referenced to behavioral models in Deliverable #3.

3g. Budget Constraints

Content

This section shows the budget for the project, expressed in money or available resources. For AUGMS, the total budget is EGP 500,000, sourced from the university's infrastructure fund, covering software development (EGP 300,000), integration/testing (EGP 150,000), and training (EGP 50,000). No additional hardware purchases are allowed.

Motivation

The requirements must not exceed the budget. This limitation may constrain the number of requirements that can be included in the product. The intention of this question is to determine whether the product is really wanted.

Considerations

The intention is to restrict the wildest ambitions and to prevent the team from gathering requirements for an Airbus 380 when the budget can buy only a Cessna.

Is it realistic to build a product within this budget? If the answer to this question is no, then either the client is not really committed to building the product or the client does not place enough value on the product. In either case you should consider whether it is worthwhile continuing.

Form

A written statement giving the amount of the budget and the source of the funding. For AUGMS, a breakdown table aligned with CRUD coverage in Deliverable #3.

3h. Enterprise Constraints

Content

This section contains requirements that are specific to the enterprise that is making the investment in your project. For AUGMS, the system must comply with university IT policies (e.g., on-premise only, no cloud) and use only approved components (e.g., Windows servers). It must also make all reports available to the University President.

Motivation

To understand requirements that sometimes appear irrelevant or irrational because they are not obviously relevant to the goals of the project.

Examples

The product shall be installed using only American-made components. The product shall make all functionality available to the CEO.

Considerations

Did you intend to develop the product on a Macintosh, when the office manager has laid down an edict that only Windows machines are permitted? Is a director also on the board of a company that manufactures products similar to the one that you intend to build? Whether you agree with these enterprise requirements has little bearing on the outcome. The reality is that the system has to comply with enterprise requirements even if you can find a better, more efficient, or more economical solution. A few probing questions here may save some heartache later. The enterprise requirements might be purely concerned with the politics inside your organization. However, in other situations you may need to consider the politics inside your customers' organizations or the national politics of the country. Another way to think about the enterprise requirements is that they are constraint requirements that have been defined by strategic decisions that are outside the obvious boundary of your project scope.

Form

A list of constraints with rationale, cross-referenced to Section 3a Solution Constraints.

4. Naming Conventions and Terminology

It has been our experience that all projects have their own unique vocabulary usually containing a variety of acronyms and abbreviations. Failure to understand this project-specific nomenclature correctly inevitably leads to misunderstandings, hours of lost time, miscommunication between team members, and ultimately poor-quality specifications.

4a. Definitions of All Terms, Including Acronyms, Used by Stakeholders Involved in the Project

Content

A glossary containing the meanings of all names, acronyms, and abbreviations used by the stakeholders. Select names carefully to avoid giving a different, unintended meaning. If the work that you are studying already has a glossary of terms then use this as your starting point. This glossary should be enlarged and refined as the analysis proceeds, but for the moment, it should introduce the terms that the stakeholders use and the meanings of those terms. This glossary reflects the terminology in current use within the work area. You might also get started by building on the standard names used within your industry. For each term, write a description. The appropriate stakeholders must agree on this description of the meaning of the term. We suggest you add all acronyms and abbreviations. We often encounter situations where team members use acronyms, but admit they do not know the meanings of those acronyms. This section gives you a place to register your acronyms. For AUGMS: AUGMS: Automated University Garage Management System – the product being developed. LPR: License Plate Recognition – technology for automatic vehicle identification. EV: Electric Vehicle – refers to charging services offered. CRUD: Create, Read, Update, Delete – matrix used for validation (see Deliverable #3).

Motivation

Names are very important. They invoke meanings that, if carefully defined, can save hours of explanations. Attention to names early in the project helps to highlight misunderstandings. As the detailed work progresses the glossary provides input to the more precisely specified business/work data model and data dictionary – see section 7 of the template. As the analysis data dictionary evolves, many of the definitions from the glossary are expanded in the dictionary by adding their data composition.

Considerations

Make use of existing references and existing data dictionaries. Obviously, it is best to avoid renaming existing items unless they are so ambiguous that they cause confusion. From the beginning of the project, emphasize the need to avoid homonyms and synonyms. Explain how they increase the cost of the project.
Form An existing glossary of terms, or a pointer to industry dictionaries, or a list of terms commonly used in the problem domain along with a sentence describing the meaning and purpose of each term. For AUGMS, this builds on the refined domain classes from Deliverable #3 (Section I.1.2).

5. Relevant Facts and Assumptions

Relevant facts are external factors that have an effect on the product but are not covered by other sections in the requirements template. They are not necessarily translated into requirements but could be. Relevant facts alert the developers to conditions and factors that have a bearing on the requirements.

5a. Relevant Facts

Content

Relevant facts are external factors that have an effect on the product but are not covered by other sections in the requirements template. They are not necessarily translated into requirements but could be. Relevant facts alert the developers to conditions and factors that have a bearing on the requirements. For AUGMS: Facts: Garage capacity is 1,200 spots; peak usage 08:00–10:00 with 650 vehicles/hour; Egyptian license plates follow a specific format (3 letters + 4 digits). Assumptions: Network bandwidth will support real-time sensor updates; users have university-issued emails for notifications.

Motivation

To alert developers to conditions, factors, or issues that might have an effect on the product requirements.

Considerations

Relevant facts might be the speed of the train, the government regulations covering the securities business, the liquidity of the stock market, the number of ice cream flavors you can buy, and so on. Relevant facts are often concerned with external (to the organization) conditions over which you have little control.

Form

A simple list of facts and assumptions, potentially in a table format separating facts from assumptions.

.

6. The Scope of the Work

The scope of the work determines the boundaries of the business area to be studied and outlines how it fits into its environment. Once you understand the work and its constraints, you can establish the scope of the product see Section 8 of the template.

6a. The Current Situation

Content

This is an analysis of the existing business processes, including the manual and automated processes that might be replaced or changed by the new product. In terms of the Volere Brown Cow model you refer to this view as the “How Now” view. Business analysts might already have done this investigation as part of the business case analysis for the project. This is where it might be appropriate to build some Business process models. These are models of the processes that the business uses to carry out the work of the organization. The models include roles, individuals, departments, technology and procedures. They illustrate the workflow and the dependencies between the components of the process.

Motivation

If your project intends to make changes to an existing manual or automated system, you need to understand the effect of proposed changes. The study of the current situation provides the basis for understanding the effects of proposed changes and choosing the best alternatives. Business process modelling does not always lead to building software. Instead, some changes in procedures and the way roles are allocated might be the best way of making a necessary improvement.

Form

There are many different notations suitable for building business process models, for example: activity diagrams, business process diagrams, swimlane diagrams, dataflow diagrams.

6b. The Context of the Work

Content

The work context diagram identifies the boundaries of the work that you need to investigate to be able to build the product. Note that it includes more than the intended product. Unless you understand the work that the product will support, you have little chance of building a product that will fit cleanly into its environment. The adjacent systems on the example context diagram (e.g., Weather Forecasting Service) indicate other subject matter domains (systems, people, and organizations) that need to be understood. The interfaces between the adjacent systems and the work context indicate why we are interested in the adjacent system. In the case of Weather Forecasting Service, we can say that we are interested in the details of when, how, where, who, what, and why it produces the District Weather Forecasts information. For AUGMS, context includes users, sensors, cameras, ID system – diagram with flows like Vehicle Details in, Entry Log out.

Motivation

To clearly define the boundary for the study of the work and hence the requirements effort. Without this definition, we have little chance of building a product that will fit seamlessly into its environment.

Examples

This work context model defines the connections between the part of the world that is under investigation and other people, organizations, hardware and software (referred to as adjacent systems). The inputs and outputs represent the data and material that travels between the work and other parts of the world. The work context is the basis for partitioning the investigation and discovering the requirements.

Considerations

The names used on the context diagram should be consistent with the naming conventions (section 4) and should eventually be defined in the data dictionary (section 7). Without these definitions, the context model lacks the required rigor, and it may be misunderstood. Relevant stakeholders must agree to the definitions of the interfaces shown on the context model.

Form

A diagram showing the inputs and outputs that flow between the work and the adjacent systems. or A table that identifies all the inputs and outputs that flow between the work and the adjacent systems. The names of the inputs and outputs are eventually defined in the data dictionary – see section 7b. For AUGMS, context diagram as in examples, with adjacent systems like ParkingUser, Sensor.

6c. Work Partitioning

Content

A list showing all business events to which the work responds. Business events are happenings in the real world that affect the work. They also happen because it is time for the work to do something—for example, produce weekly reports, remind non-paying customers, check the status of a device, and so on. The response to each event is called a business use case (known as a BUC); it represents a discrete partition of work that contributes to the total functionality of the work. The event list includes the following elements:

- Event name
- Input or triggering data flow from adjacent systems (identical with name on context diagram)
- Output/s to adjacent systems (identical with name/s on context diagram)
- Brief summary of the business use case (This is optional, but we have found it is a very useful first step in defining the requirements for the business use case—you can think of it as a mini-scenario.)
- Classes of business data relevant to this event (you won't know this early in the study of the event, as you go into detail you will start to understand the essential data and you can add it to the event list.) For AUGMS, events like Vehicle Arrives, Service Requested – from Deliverable #2 Event Table.

Motivation

To identify logical chunks of the work that can be used as the basis for discovering detailed requirements. These business events also provide the subsystems that can be used as the basis for managing detailed analysis and design. Each business event has a business use case (BUC) whose details can be studied independently. However all BUCs connect to each other through the stored business data (see section 7).

Examples

Business Event List Event Name | Input and Output | Summary of BUC

1. Weather Station transmits reading | Weather Station Readings (in) | Record the readings as belonging to the weather station. [And so on for 11 events in template example.]

Considerations

Attempting to list the business events and do a one-sentence summary of each of the BUCs is a way of testing the work context. This activity uncovers uncertainties and misunderstandings about the project and facilitates precise communications.

When you do an event analysis, it will usually prompt you to make some changes to your work context diagram. We suggest you gather requirements for discrete sections of the work. This requires you to partition the work, and we have found business events to be the most convenient, consistent, and natural way to break the work into manageable units and to be able to trace the details back to the scope of the work.

Form

Business event list/table containing for each event: Event number, Event name, Name of input, Name of output/s, Summary of the business event response. The names on the business event list must match the names on the work context model/table ref. 6.b. For AUGMS, table from Deliverable #2.

6d. Specifying a Business Use Case (BUC)

Content

A specification of the details of how a Business Use Case (BUC) responds to a Business Event. For AUGMS, each BUC like "Validate Vehicle Entry" is specified using Activity Diagrams from Deliverable #3.

Motivation

To understand the detailed business response that must be carried out when a business event takes place and provide a basis for discovering the detailed requirements. The understanding of the BUC also provides the basis for discussing which parts of the BUC should be carried out by the product that will be built.

Examples

In the sample specifications included with the download of this template you will find examples of BUC scenarios.
Considerations
Whatever approach you use to specify the details of a BUC, you should stay within the boundary of the input and output/s for that business event.

If you discover additional input or output data then it is an indication that you need to make changes to the input/output data on the event list and also on the work context diagram.

Form

A BUC can be specified using any combination of models that suits the analyst. The most common approaches are: activity diagrams, BUC scenarios, process flow diagrams, sequence diagrams, mind maps, interview notes.... The only caveat is that the inputs and outputs on your BUC are precisely the same and hence traceable to the inputs and outputs on the corresponding Business Event. For AUGMS, Activity and Sequence Diagrams from Deliverable #3.

7. Business Data Model and Data Dictionary

7a. Business Data Model

Content

A specification of the essential subject matter, business objects, entities, and classes that are germane to the product. It might take the form of a first-cut class model, an entity-relationship model, or any other kind of data model. For AUGMS, UML Class Diagram with classes like User, Vehicle, ParkingSpot (from Deliverable #3 Section I.1.3).

Motivation

To clarify the system's subject matter, thereby triggering recognition of requirements not yet considered. To discover missing requirements you can cross check the data model and the events using a Create, Reference, Update, Delete (CRUD) table. The data model is a specification for all of the business data that is relevant to the scope of the work.

Examples

This is a model of the business system's business subject matter using the Unified Modelling Language (UML) class model notation. This is all the data that is Created, Referenced, Updated and Deleted by processes within the scope of the work being studied. See section 6 for more about the scope of the work.

Considerations

Are there any data or object models for similar or overlapping systems that might be a useful starting point? Is there a domain model for the subject matter dealt with by this system?

Form

There are many different types of data models that you can use to model the business data. The ones you are most likely to come across are:

- UML class model
- Crow's foot diagrams
- Entity Relationship diagrams
- A table showing: Class Name, Relationships between classes, Attributes for each class. If your organization prefers a particular model then you must use that one. The important thing is that the data model that you build is a business data model, not a design for a database. Your model is concerned with identifying business classes by making a logical partitioning of all the data within the work context and the necessary business relationships between those business classes. Your model is used as input to designing how the data will be implemented. The definitions of the attributes in each business class is in the data dictionary (see section 7b). For AUGMS, Mermaid syntax diagram from Deliverable #3.

7b. Data Dictionary

The glossary described earlier in section 4 of the template is the starting point for establishing common understanding of terminology. As you start to define the scope of the investigation you define the data inputs and outputs in a formal data dictionary. The terms that you define in this dictionary, right down to elemental level, are the same terms that you use when defining detailed atomic requirements.

The data dictionary specifies the content of:

- Classes on the data model
- Attributes of the classes
- Relationships between the classes
- Inputs and Outputs on all models
- Elements of data within the Inputs and Outputs When implementation decisions are made the technical specifications for the interfaces should be added to the dictionary. For AUGMS, dictionary entries like Vehicle = *License Plate* + *Make* + *Model*, etc

.

Motivation

The work context diagram provides an accurate definition of the scope of the work being studied and the product scope diagram (See Sections 8a & 8b) defines the boundary of the product to be built. These definition can be completely accurate only if the information flows bordering the scope have their attributes defined.

Examples

The following is a partial data dictionary for the road de-icing project we have been using as an example in this template. Note that this version of the dictionary is sorted alphabetically within Type. When implementation decisions are eventually made the format of the data is added to the dictionary by the designers/implementors. [Table with Name, Content, Type]

Considerations

The dictionary provides a link between the requirements/business analysts and the designers/developers/implementers. The implementers add implementation details to the terms in the dictionary, defining how the data will be implemented. Also, implementers add terms that are present because of the chosen technology and that are independent of the business requirements. As you study the work you often discover that an entry that you have put in the Naming Conventions and Terminology (section 4) turns out to be a specific flow of data or an attribute data. When this happens you should transfer the entry to the data dictionary.

Form

The data dictionary is maintained in a variety of forms, depending on the tools that you have at your disposal. The important issue is that you make it as easy as possible to cross reference the use of terms in requirements, documents and models with their definitions in the dictionary. Common forms for maintaining the data dictionary are spreadsheets, databases and automated requirements tools. For AUGMS, spreadsheet table.

8. The Scope of the Product

8a. Product Boundary

A use case diagram identifies the boundaries between the users (actors) and the product. You arrive at the product boundary by inspecting each business use case and determining, in conjunction with the appropriate stakeholders, which part of the business use case should be automated (or satisfied by some sort of product) and what part should be done by the user or some other product. This task must take into account the abilities of the users/actors (section 2), the constraints (section 3), the goals of the project (section 1), and your knowledge of both the work and the technology that can make the best contribution to the work. The example use case diagram shows the users/actors outside the product boundary (the rectangle). The product use cases (PUCs) are the ellipses inside the boundary. The numbers link each PUC back to the BUC that it came from (see section 7). The arrows denote usage. In this version of a PUC diagram we put names on the arrows to make it more precise and traceable. Note that actors can be either automated or human. You derive the PUCs by deciding where the product boundary should be for each business use case (BUC). These decisions are based on your and appropriate stakeholders' knowledge of the work and the requirements constraints. Note that the PUCs that you come up with must be traceable back to the BUCs. When you implement a PUC your internal design might, and probably will, mean that you will implement a PUC with several system use cases (SUCs). For AUGMS, updated Use Case Diagram from Deliverable #3 (Section II), with actors like ParkingUser, GarageAdmin.

Motivation

A use case diagram identifies the boundaries between the users (actors) and the product. You arrive at the product boundary by inspecting each business use case and determining, in conjunction with the appropriate stakeholders, which part of the business use case should be automated (or satisfied by some sort of product) and what part should be done by the user or some other product. This task must take into account the abilities of the users/actors (section 2), the constraints (section 3), the goals of the project (section 1), and your knowledge of both the work and the technology that can make the best contribution to the work.

Examples

The numbers on the PUC diagram above correspond to the BUC numbers on the Business Event List (see section 7).

Form

A product use case diagram. A product scope diagram supported by individual use case specification. Specifications and prototypes of the interfaces on the scope diagram. For AUGMS, UML Use Case Diagram.

8b. Product Use Case Table

Content

The product scope diagram is a useful summary of all the interfaces between the product and other automated systems, organizations and users. If there are more than around 20 PUCs, the diagram becomes too large and a table is better. The table lists PUCs with brief descriptions, actors, and links to BUCs. For AUGMS, table with PUC ID, Name, Actor – e.g., PUC-01 Register Vehicle, ParkingUser.

Motivation

The product scope diagram is a useful summary of all the interfaces between the product and other automated systems, organizations and users. If there are more than around 20 PUCs, the diagram becomes too large and a table is better.

Form

Product Use Case Table with PUC number, Name, Actor, Summary, Linked BUC. For AUGMS, from Deliverable #2 refined in #3.

9. Functional Requirements

Content

A specification of what the product must do. For AUGMS, atomic functional requirements like F-01: System shall recognize license plate in ≤2 seconds.

Motivation

To define the product's capabilities.

Examples

The product shall recognize a license plate and open the gate in ≤ 2 seconds

Considerations

The functional requirements detail the business solution. It is at this point that you completely describe the product to be built by describing its functional requirements. You can use a number of different models to complement the requirement descriptions. This combination ensures that the developers working on the product have a complete understanding of what they have to build.

Form

List of atomic functional requirements with description, rationale, fit criterion.

Cross-reference to PUCs. For AUGMS, table with Req ID, Description, Fit Criterion.

Non-functional Requirements

The following sections 10-17 describe the non-functional requirements. The form of these requirements is the same as for the functional requirements as described above.

10. Look and Feel Requirements

Content

AUGMS must present a clean, modern, and intuitive interface for students, faculty, and admins. The system's visual style should reflect the university's identity and maintain a consistent design across all components.

Requirements

- UI must follow a minimal, modern design with consistent color schemes, icons, and typography.
- The interface must look clean and uncluttered, prioritizing essential parking information and user actions.
- Admin dashboard should use structured layouts: tables, charts, color-coded indicators (e.g., green for available, red for full).
- Visual indicators must be used instead of text wherever possible (e.g., icon for EV charger, maintenance, reservation).
- The system must adapt properly to desktop, tablet, and mobile screens (responsive design).
- Color choices must respect accessibility standards:
  - Contrast ratio ≥ 4.5:1
  - Colorblind-friendly palette
- The interface must feel consistent across:
  - Student portal
  - Faculty portal
  - Admin dashboard
- Camera feed windows, sensor status indicators, and occupancy levels must use clear visual coding (e.g., floor color mapping).

Fit Criterion

Usability testing shows that at least 90% of test users can complete core tasks (register vehicle, check parking availability, submit service request) without external guidance.

11. Usability and Humanity Requirements

Content

AUGMS should be easy to learn, simple to operate, and require minimal training for both users and admins.

Requirements

- The student/faculty portal must be fully usable without reading documentation.
- Key actions must be achievable in 3 clicks or fewer, such as:
    - Registering a vehicle
    - Requesting a service
    - Viewing garage occupancy
- Navigation must be simple with a clear menu structure: Dashboard → Vehicle → Services → Account.
- Admin dashboard shall include:
    - Search bar for fast user lookup
    - Real-time occupancy widget
    - Alerts panel for sensor/camera issues
- Text labels must be simple, direct, and jargon-free.
- Error messages must clearly explain the issue and how to fix it (e.g., "Plate number already registered. Please update your vehicle info.").

- The system must support Arabic and English, with identical functionality across both languages.
- Forms must include validation to prevent errors (e.g., invalid plate format, missing fields).
- Pages must load within ≤ 2 seconds under normal conditions.
- Accessibility:
    - Screen reader support
    - Keyboard navigation shortcuts
    - Tooltips for icons
    - Large clickable areas for mobile use

Fit Criterion

User experience evaluation scores ≥ 4.2/5 across usability metrics (clarity, ease of use, task completion speed, satisfaction).

12. Performance Requirements

Content

AUGMS must operate efficiently even under peak traffic conditions, ensuring fast response times, accurate sensor updates, and reliable ANPR processing.

Requirements

System Response & Speed

- Dashboard load time (student/faculty): ≤ 2 seconds.
- Admin dashboard load time: ≤ 3 seconds due to analytics.
- Vehicle registration submission: Processed within ≤ 1 second.
- Parking occupancy updates:

        - Sensor data reflected on UI within ≤ 5 seconds.
        - ANPR entry/exit processing within ≤ 3 seconds.

Peak Load Requirements

- Must support 5,000 active users checking availability simultaneously.
- Must handle 200 concurrent ANPR recognitions per hour without failure.
- Database must support:
    - 100+ simultaneous read/write operations
    - Non-blocking queries during peak load

Scalability

- Able to scale horizontally by adding new sensors, additional garage floors, or new zones without major redesign.
- Able to handle future integrations (EV chargers, smart gates, new IoT devices).

Reliability

- System uptime must meet 99% during university working days.
- Automatic failover for hardware failures (sensor offline → manual admin approval mode).
- Automatic retries for dropped sensor reports.

Storage Performance

- ANPR images stored with compression and must retrieve in ≤ 2 seconds.
- Logs must support 30 days of history without degradation.

Fit Criterion

Performance testing demonstrates the system meets all time-based thresholds under simulated peak load with ≤ 5% deviation.

13. Operational and Environmental Requirements

Content

The environment in which AUGMS will operate includes an enclosed multilevel garage structure, automated entry/exit gates, license plate recognition cameras, and IoT parking sensors. The system interacts with both physical hardware and the university's internal network.

Requirements

- The system must remain operational 24/7, supporting continuous garage entry, exit, and monitoring.
- All sensor equipment, including ANPR cameras, must operate reliably in varying environmental conditions:
    - Temperature range: –10°C to 55°C
    - Humidity: up to 95% non-condensing
    - Exposure to dust, fog, and low light
- Hardware must automatically reconnect to the system within 60 seconds after power outages or network disruptions.
- In case of hardware failure (camera offline, sensor malfunction), the system must fall back to *manual override mode* where admins **can approve** entries/exits manually.
- The system must operate efficiently during peak hours:
    - Morning rush (7:00–10:00 AM)
    - Midday peak (12:00–2:00 PM)
    - Evening exit (4:00–6:00 PM)
- The database server must be hosted on university infrastructure and support concurrent access from up to 5,000 users.
- The system must support integration with future hardware (new sensors, IoT devices) without requiring major architectural change**.**

Fit Criterion

- 99% uptime during academic operation hours
- Recovery from network outage within ≤ 60 seconds
- ≤ 2% sensor error rate during operating hours

14. Maintainability and Support Requirements

Content

AUGMS must be easy to maintain, extend, update, and troubleshoot by IT staff and developers.

Requirements

- Codebase must follow modular architecture:
    - Separate modules for user management, parking monitoring, ANPR, admin services, and reporting
- System logs must include:
    - Entry attempts
    - Exit attempts
    - Sensor failures
    - Admin overrides
    - Error messages with timestamps
- Admins must be able to:
    - Disable or bypass malfunctioning sensors
    - Add new floors/zones
    - Edit capacity of garage sections
    - Replace a camera or sensor ID without system reinstallation
- Updates must allow rolling deployment, meaning non-critical updates can be applied without shutting down the entire system.

  Fit Criterion

  Routine maintenance (logs check, sensor replacement, recalibration) must be completable within 20 minutes.

15. Security Requirements

Content

AUGMS stores personal information (name, plate number, vehicle data), which demands strong protection.

Requirements

- Authentication must enforce university Single Sign-On (SSO) OR username/password with complexity rules.
- All plate numbers, user data, and logs must be AES-256 encrypted in storage.
- All data transfer must use HTTPS with TLS 1.3.
- Access levels:
  - Student/Faculty: Personal dashboard only
  - Admin: Approvals, occupancy, service handling
  - Management: Reports and analytics
- System must lock the account after 5 failed login attempts for 30 minutes.
- Audit logs must record all admin-level actions (approval, deletion, override).
- Database must restrict direct access to DBAs only.

Fit Criterion

Security audit from IT department shows *zero critical vulnerabilities*.

16. Cultural Requirements

Content

The university population includes Arabic and English speakers; UI must be inclusive and culturally neutral.

Requirements

- All user-facing text must be available in English and Arabic.
- The system must avoid:
  - Cultural bias
  - Region-specific idioms
- Icons and visual indicators must follow international standards (ISO Parking Symbols).
- Date and time formatting must support:
  - 24-hour format by default
  - Localized formats for reports if needed

Fit Criterion

User acceptance testing shows ≥ 90% comprehension for both English and Arabic users.

17. Legal Requirements

Content

AUGMS must comply with privacy, security, and data management laws.
Requirements

- Data handling must follow GDPR-like principles:
  - Collect only data required for operation
  - Delete or anonymize older logs after the retention period (e.g., 30 days for ANPR logs)
- The system must provide the ability for users to request removal of their vehicle registration data if they leave the university.
- All audit logs must comply with university IT governance and data retention policies.
- System must comply with WCAG 2.1 AA accessibility guidelines.
- Personal data must never be shown to unauthorized individuals.

  Fit Criterion

  Compliance checks show no violations of university or national data privacy laws.

Project Issues

The following *sections 18-27* contain issues that must be faced if the requirements are to be met and the product to become a reality. These sections also connect the requirements with the project activities that discover and progress the requirements. If you are using a consistent language for communicating requirements then project managers can use the requirements as input to steering the project. The Volere Requirements Knowledge Model (included with the download of Version 16 of the template) provides the basis for a requirements common language by identifying classes of requirements knowledge and the relationships between them. Each of the classes of knowledge is cross-referenced to sections in this template.

18. Open Issues

Issues that have been raised and do not yet have a conclusion.

- • Should users receive notifications via email, SMS, or push notifications?
- • Should users be able to reserve a parking spot in advance?
- • Will the system integrate with existing student ID card systems?
- • How long should ANPR images be stored before deletion?
- • Should service payment handling be integrated (online payments)?
- • What service types will be available beyond EV charging and cleaning?

19. Off-the-Shelf Solutions

Potential ready-made modules AUGMS may incorporate:

Hardware Solutions

- ANPR cameras (HikVision, Axis)
- Ultrasonic or infrared parking sensors
- Pre-built smart gate systems

Software Solutions

- Parking management dashboards
- BI tools for reporting (Power BI, Tableau)
- Cloud logging systems

Benefits

- Reduced development time
- Higher reliability on tested hardware
- Easier integration with university infrastructure.

20. New Problems

Introducing AUGMS creates new risks:

- Increased operational dependency on sensors and network stability

- Security/privacy concerns over license-plate surveillance
- Higher maintenance complexity (IoT hardware needs upkeep)
- Potential dissatisfaction from users if sensors malfunction
- Cost of updating or replacing hardware over time

21. Tasks

Detailed task list to complete the system:

Planning

- Finalize requirements
- Establish architecture and infrastructure

Development

- Create student/faculty portal
- Build admin dashboard
- Implement ANPR integration
- Implement real-time occupancy logic
- Build service request workflow
- Build reporting engine

Deployment

- Install sensors and cameras
- Connect hardware to network
- Deploy backend and frontend
- Configure failover system**s**

Testing

- Unit, integration, system testing
- Stress test with simulated peak load
- Security penetration testing
- UAT with students, faculty, and admins

  Training
- Train garage administrators
- Provide user documentation
- Create video tutorials

22. Migration to the New Product

Migration Steps

- Import existing parking data (if any)
- Install hardware before enabling the system
- Begin with one floor (pilot phase), then scale
- Maintain manual override for at least 30 days after launch
- Train users with portal guides

Data Migration

- Clean old data to match new model
- Map legacy vehicle IDs to new user accounts
- Validate imported records before enabling ANPR

23. Risks

- **Technical Risks**
- **Sensor or camera hardware failure**
- **ANPR accuracy issues due to dirt, weather, lighting**
- **Network downtime affecting gate responsiveness**
- **Operational Risks**
- **Unauthorized access attempts**
- **Misuse of service request system**
- **Admin errors during manual override**
- **Business Risks**
- **Budget overruns for hardware**
- **Resistance from staff or users unwilling to adopt new tech**
- **High maintenance cost if equipment is damaged frequently**

24. Costs

Cost Components

- **Hardware:**
    - ANPR cameras
    - Gate controllers
    - Parking sensors
    - Network equipment
- **Software Development:**
    - Web portal
    - Integration modules
    - Reporting & analytics
- **Operation:**
    - Server hosting
    - Energy consumption
    - Routine maintenance
- **Personnel:**
    - Developers
    - Technicians
    - Admin training
- **Contingency Cost:**
    - Hardware replacement reserve

25. User Documentation and Training

User Documentation

- Quick-start guide for students/faculty
- How to register a vehicle
- How to request services
- How to check parking availability

Admin Documentation

- Full admin manual
- Sensor troubleshooting guide
- How to approve or reject registrations
- How to override gate operations
- Error handling and escalation procedures

Training Methods

- In-person training sessions for admins
- Short tutorial videos for users
- Knowledge base articles on the university website

26. Waiting Room

Non-priority features (future releases):

- Mobile app with real-time alerts
- Parking spot reservation system
- Dynamic pricing based on demand
- AI predicting availability patterns
- Employee parking zones with priority access
- Smart navigation to nearest available spot
- Parking heatmap visualization

27. Ideas for Solutions

Architectural Ideas

- Microservices architecture for scalability
- IoT gateway to handle sensor communication
- Cloud-based reporting engine
- AI model to detect abnormal behavior (fraud, tailgating)

UX Solutions

- Clean dashboard with color-coded indicators
- Notifications for available spots
- Live camera feed for admins

Operational Ideas

- Scheduled automated health check for all sensors
- Self-calibrating ANPR cameras
- Predictive maintenance alerts