Epoka Unified System

Requirements Specification

Version 1.0

April 02, 2019

Megi Hoxha* Ergi Dervishaj* Gerard Kraja* Krist Kokali* Paolo Miraka*

v

Table of Contents

1. Executive Summary

1.1 Project Overview

Epoka Unified System is a web application intended to provide an easy-to-use, intuitive interface, which allows the users to access their university related data such as courses, attendance etc. This application also provides a channel of communication between the student and the professor.

The intended audience of EUS are:

- Students
- Professors

1.2 Purpose and Scope of this Specification

This document specifies the requirements of the project without any technical implementation, thus focusing solely on the description of the entire infrastructure of the program.

In scope

- Modification of the methods of collecting and storing information to meet the rules of GDPR set by the European Union.
- Modification of Labor Relations Processing to meet the requirements of the Albanian legislation regarding the topic.

Out of Scope

- Technical implementation of the system.
- Development obstacles and solutions.
- The way of obtaining and using the software.

2. Product/Service Description

The Epoka-Unified-System is meant to be used by students and professors which is the main idea behind the designation of the entire solution. The whole student and professor experience in Epoka University is a continuous series of exams, project and assignment deadlines which have to be respected. At times it is pretty hard to remember every single task or exam that the student or professor has to participate in. This unanimously came to be the design philosophy behind every decision that the team is taking. Practically the team is attempting to provide a simple and clean interface that displays efficiently and with no delay all the personal data of a student or professor. To further simplify things the addition of notifications and reminders might help increase the productivity and subsequently the performance of the users of the system.

2.1 Product Context

How does this product relate to other products? Is it independent and self-contained? Does it interface with a variety of related systems? Describe these relationships or use a diagram to show the major components of the larger system, interconnections, and external interfaces.

2.2 User Characteristics

There are two user categories who will be using this product:

- Students
    I. Are able to use the GUI to display the needed information.
    II. Are able to download, install and keep the application updated so it functions correctly at all times.

I. Tend to use the Electronic System at times where it is crucial to obtain the necessary information as fast as possible.
II. Will rely on the displayed data to do many crucial university-related actions.

- Professors
  I. Have average to expert level technical knowledge of a system of the sort.
  II. Repeatedly communicate with the system to provide the grades and assignments in a timely manner.
  III. Rely on the communication bridge (EIS system currently) to correctly and timely transmit grades and assignments to the students.

2.3 Assumptions

The EUS system is designed to provide cross-platform functionality because of it's web-based structure. User's are expected to run the software on the well known platforms Windows, Mac and mobile environments. Because of the complexity of the solution the device should have at least average computation power and a reliable internet connection. The intended audience (students and professors) are expected to be at least semi proficient in the English language to understand the utilities provided. The interface is being developed to be simplified and the modules are designed to effectively group together functionalities to make the learning curve of the program as smooth as possible.

2.4 Constraints

Describe any items that will constrain the design options, including

- parallel operation with an old system
- audit functions (audit trail, log files, etc.)
- access, management and security
- criticality of the application
- system resource constraints (e.g., limits on disk space or other hardware limitations)
- other design constraints (e.g., design or other standards, such as programming language or framework)

2.5 Dependencies

- The product requires a stable internet connection at all times.
- The system relies heavily on the professors which provide the timetable, grades and assignments for the students.
- Professor views and functionalities must function correctly first to provide the server information which will adequately be displayed to the students.
- The back-end structure should be ready to respond to all requests at any time which might be highly concurrent peaking at University operating hours.
- The EUS is a very complex system which manages crucial operations for students and professors therefore the administrators should always be available to respond to complaints when certain scenarios fail to provide functionality or a better way is suggested.

3. Requirements

- Describe all system requirements in enough detail for designers to design a system satisfying the requirements and testers to verify that the system satisfies requirements.
- Organize these requirements in a way that works best for your project. See Appendix DAppendix D, Organizing the Requirements for different ways to organize these requirements.
- Describe every input into the system, every output from the system, and every function performed by the system in response to an input or in support of an output. (Specify what functions are to be performed on what data to produce what results at what location for whom.)
- Each requirement should be numbered (or uniquely identifiable) and prioritized.
  See the sample requirements in Functional Requirements, and System Interface/Integration, as well as these example priority definitions:

Priority Definitions

The following definitions are intended as a guideline to prioritize requirements.

- Priority 1 – The requirement is a “must have” as outlined by policy/law
- Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
- Priority 3 – The requirement is a “nice to have” which may include new functionality

It may be helpful to phrase the requirement in terms of its priority, e.g., "The value of the employee status sent to DIS **must be** either A or I" or "It **would be nice** if the application warned the user that the expiration date was 3 business days away". Another approach would be to group requirements by priority category.

- A good requirement is:
  - Correct
  - Unambiguous (all statements have exactly one interpretation)
  - Complete (where TBDs are absolutely necessary, document why the information is unknown, who is responsible for resolution, and the deadline)
  - Consistent
  - Ranked for importance and/or stability
  - Verifiable (avoid soft descriptions like “works well”, “is user friendly”; use concrete terms and specify measurable quantities)
  - Modifiable (evolve the Requirements Specification only via a formal change process, preserving a complete audit trail of changes)
  - Does not specify any particular design
  - Traceable (cross-reference with source documents and spawned documents).

3.1 Functional Requirements

In the example below, the requirement numbering has a scheme - BR_LR_0## (BR for Business Requirement, LR for Labor Relations). For small projects simply BR-## would suffice. Keep in mind that if no prefix is used, the traceability matrix may be difficult to create (e.g., no differentiation between '02' as a business requirement vs. a test case)

The following table is an example format for requirements. Choose whatever format works best for your project.

For Example:

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_LR_05	The system should associate a supervisor indicator with each job class.	Business Process = “Maintenance	3	7/13/04	Bob Dylan, Mick Jagger
BR_LR_08	The system should handle any number of fees (existing and new) associated with unions.	Business Process = “Changing Dues in the System” An example of a new fee is an initiation fee.	2	7/13/04	Bob Dylan, Mick Jagger

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_LR_10	The system should capture and maintain job class status (i.e., active or inactive)	Business Process = "Maintenance" Some job classes are old and are no longer used. However, they still need to be maintained for legal, contract and historical purposes.	2	7/13/04	Bob Dylan, Mick Jagger
BR_LR_16	The system should assign the Supervisor Code based on the value in the Job Class table and additional criteria as specified by the clients.	April 2005 – New requirement. It is one of three new requirements from BR_LR_03.	2		
BR_LR_18	The system should provide the Labor Relations office with the ability to override the system-derived Bargaining Unit code and the Union Code for to-be-determined employee types, including hourly appointments.	April 2005 – New requirement. It is one of three new requirements from BR_LR_04. 5/11/2005 – Priority changed from 2 to 3.	~~2~~ 3		

3.2 Non-Functional Requirements

3.2.1 User Interface Requirements

In addition to functions required, describe the characteristics of each interface between the product and its users (e.g., required screen formats/organization, report layouts, menu structures, error and other messages, or function keys).

3.2.2 Usability

Include any specific usability requirements, for example,

Learnability

- The user documentation and help should be complete
- The help should be context sensitive and explain how to achieve common tasks
- The system should be easy to learn

(See http://www.usabilitynet.org/)

3.2.3 Performance

Specify static and dynamic numerical requirements placed on the system or on human interaction with the system:

- Static numerical requirements may include the number of terminals to be supported, the number of simultaneous users to be supported, and the amount and type of information to be handled.
- Dynamic numerical requirements may include the number of transactions and tasks and the amount of data to be processed within certain time period for both normal and peak workload conditions.

All of these requirements should be stated in measurable form. For example, "95% of the transactions shall be processed in less than 1 second" rather than "an operator shall not have to wait for the transaction to complete".

3.2.3.1 Capacity

Include measurable capacity requirements (e.g., the number of simultaneous users to be supported, the maximum simultaneous user load, per-user memory requirements, expected application throughput)

3.2.3.2 Availability

Include specific and measurable requirements for:

- Hours of operation
- Level of availability required
- Coverage for geographic areas
- Impact of downtime on users and business operations
- Impact of scheduled and unscheduled maintenance on uptime and maintenance communications procedures
- reliability (e.g., acceptable mean time between failures (MTBF), or the maximum permitted number of failures per hour).

3.2.3.3 Latency

Include explicit latency requirements, e.g., the maximum acceptable time (or average time) for a service request.

3.2.4 Manageability/Maintainability

3.2.4.1 Monitoring

Application Performance Monitoring:

- The performances pulse of the application will be measured by writing reports. High quality reports give additional insight into performance trends and indicate directly to the performance of the application.
- An additional way to measure the application's performance will be by allowing the users to rate the application and write their suggestions on what should be improved.

Log Monitoring:

- Log files will be generated. Errors, problems, and more information will be constantly logged and saved for analysis.
- Monitoring logs will help to identify security events that occurred or might occur.

3.2.4.2 Maintenance

- The existing software will be modified while preserving its integrity, to fulfill the objective of the software maintenance.
- To maintain the software, the collaborators will try to re-document their software systems from time to time to keep it updated.
- The users will be automatically informed in case of any error by using a message digest which is known to be relatively free from collisions.

3.2.4.3 Operations

The normal and special operations required by the user will include:

- The ability of logging in anytime and accessing their personal information.
- The users are divided into two categories (student and professor) and each one will be provided with different accessibilities.
- The personal information of the student is accessed only by them. It will include:
    - The courses they have selected to enroll to and materials regarding each course.
    - The attended/unattended hours in each one of the courses, together with the final percentage of the attendance.
    - The grades taken in each exam/quiz/homework/laboratory assignment and a prediction on the final grade of the course.

- The personal information of the professor is accessed only by them. It will include:
    - The list of the students enrolled to the course/s they lecture divided into different majors.
    - Their materials regarding the course/s.
- These accessibilities by signing in/up as a student will be:
    - **Notifications –** These will include events (ex. “The course registration week”) and deadlines (ex. “WEB Programming: Please submit the Laboratory Assignment 02 (01:45)”). The student will also be able to create a customized event/deadline by writing its name, description, also the time when it should be finished.
    - **Calendar –** It will work as an optimization of all events, deadlines and other national events (days off), all in one.
    - **Timetable –** It will include the transport and the courses timetable.

      As for the courses timetable, the timetable of the current day will be displayed first, followed by the upcoming days of the week. On the other hand, the mobile view will be a little different since the display changes. The user will have to choose the day and then the specified timetable will be shown.
    - **Courses –** During the course registration week, the student will have to choose the courses by selecting them in a list where will be displayed all the courses of its major from the first to the third year and then a request for acceptance will be sent to the professor.

      My courses – This will include me main information the student will have, including grades, course materials and attendance.
    - **My Profile –** It will have the credentials of the students, the transcription of its grades, also the GPA of them.
- These accessibilities by signing in/up as a professor will be:
    - The acceptance of the students that want to enroll to their course/s.
    - They should update the attendance for each student.
    - Create deadlines for assignment submissions.

- Add/Edit/Delete students' grades.
- Add/Edit/Delete course materials.

3.2.5 System Interface/Integration

3.2.5.1 Network and Hardware Interfaces

For the Epoka Unified System to run you will need a PC (regardless of its operating system, it can be Windows or Linux) or Mac, or on mobile (it can be Android or IOS).
Since it is not a resource hungry program, it will run on most systems without a problem.
Lastly, a functional standard keyboard and mouse is required.

3.2.5.2 Systems Interfaces

The program will be an integrated software.
There is no synchronization process since the solutions all share the same database.
This feature is one of the biggest benefits of having an integrated software solution.
The data of the application is maintained in the database (MySQL). It cannot be edited by any user.

3.2.6 Security

3.2.6.1 Protection

- The data of each user will be strictly protected.
- The personal information of the student will be protected and secured by the lecturer.
- The email and password of each user will be validated by using specific functions.
- Each lecturer will be provided with the information of the students who are enrolled to his/her course/s.
- Each student will be provided with the information related to him/her.

3.2.6.2 Authorization and Authentication

- It will be a Single-Factor Authentication, so it will rely on the email and password of the user
- The user will be authenticated by using their email and password in order for them to log in the software.
- The credentials of the user will be checked by using a function for their validation to verify their identity.
- The two different categories of the users will only be authorized with their accessibilities.
- The information of each user will be private and will be accessed only by the specific actors.
- The standard PubCookie tool will be used.

3.2.7 Data Management

- The data that this software will have to deal with varies from personal information of the student, to specific and detailed records.
- This data will be accessed and maintained by certain rules. Depending in the user's level of accessibility (student and lecturer), the range of access will be different.
- The entities and their relationships will be defined in detailed schemas and diagrams.

3.2.8 Standards Compliance

Our application is a system developed to manage all academic activity of both students and professors. Sensitive information will be stored and as such it must be protected. Given that there is existing regulation by the EU, called the General Data Protection Regulation (GDPR), we strive to comply with it in order to protect the privacy of the system users. For more information on GDPR rules and citizens rights visit this link.

3.2.9 Portability

- The system will be web-based; therefore, it will operate the same regardless of the operating system.
- The system will be programmed using technologies such as ReactJS, NodeJS, SQL, Express

3.3 Domain Requirements

- Being able to access the list of faculties and list of students of each department.
- Implementing a repository for study material sharing and assignment submission.
- Being able to access the course and transportation timetables.
- Having access to the attendance records.

4. User Scenarios / Use Cases

4.1 User Scenario

GENERAL

1 User registers – User registers with epoka email and also by filling the form
2 User login – User enters the page by entering the epoka email and the password set by the user
3 User registration/login failed – User failed to enter correct email or password
4 Check calendar – User opens the calendar to check if any upcoming event is coming soon

STUDENT

5 Update calendar – User can add a custom event to the calendar
6 Courses selection – User can select the available courses he/she wants to attend
7 Course information – User can view information about the course like attendance, grades and additional course materials
8 Timetable – User can check his/her custom timetable and also the bus timetable after selecting the courses

9 Notifications – User gets notified if a event has occurred like a homework deadline , new assignment , upcoming of a national holiday etc.

LECTURER

10 Update calendar – Lecturer can add events so individual student calendar gets updated
11 Courses selection – Lecturer select the courses he/she will lecture
12 Course information – Lecturer can add attendance and grades for individual student of the class and also add material
13 Timetable – Lecturer can edit his personal timetable and then all students custom timetable will be updated also.
14 Notifications – Lecturer can receive notifications about university event etc.

4.2 Use Cases

NR	1
Summary:	Registration procedures
Actor:	Lecturer , Student
Description:	For each level of user to enter the system, it is required that these users must register by entering their individual epoka email and password.
Precondition:	Must have epoka email address and fill in the form
Alternative:	None
Postcondition:	The user is registered and logged in the page

NR	2
Summary:	Login procedures
Actor:	Lecturer , Student
Description:	For each level of user to enter the system, it is required that these users must login by entering their individual epoka email and password.
Precondition:	User must first be registered
Alternative:	None
Postcondition:	The user can access the page and is granted their specific rights

NR	3
Summary:	Authentication failed
Actor:	Lecturer , Student
Description:	For each level of user to enter the system, it is required that these users must register by entering their individual epoka email and password.
Precondition:	Must have epoka email address and fill in the form
Alternative:	None
Postcondition:	The user is registered and logged in the page

NR	4
Summary:	Calendar interaction
Actor:	Lecturer , Student
Description:	This section is used to check all the events in a sorted way in the calendar
Precondition:	Must be successfully authenticated and clicked calendar section
Alternative:	None
Postcondition:	None

NR	5
Summary:	Custom calendar event
Actor:	Lecturer , Student
Description:	User can add a custom event to be reminded in a specific time to the calendar
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	Custom event is added

NR	6
Summary:	Courses selection
Actor:	Student
Description:	Student is shown all the courses he/she can attend and then the user can select the desired courses to be submitted to the adviser
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	Selected courses are sent to the adviser for approval

NR	7
Summary:	Course information
Actor:	Student
Description:	Student can check their attendance, grades, materials also submit work
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	None

NR	8
Summary:	Timetable
Actor:	Student
Description:	Student can check their custom timetable based on the courses they selected
Precondition:	Must be successfully authenticated and must have selected courses
Alternative:	None
Postcondition:	None

NR	9
Summary:	Notifications
Actor:	Student
Description:	Student can check any upcoming event based on the filters and the importance of the event
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	None

NR	10
Summary:	Calendar update
Actor:	Lecturer
Description:	Lecturer can add event to the calendar
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	All students based on the condition of the event will be notified

NR	11
Summary:	Course selection
Actor:	Lecturer
Description:	Lecturer can choose the classes he wants to teach
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	Timetable can be edited based on the course selection

NR	12
Summary:	Course information
Actor:	Lecturer
Description:	Lecturer can edit grades, attendance and submit additional material
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	Students gets notified on the update

NR	13
Summary:	Timetable
Actor:	Lecturer
Description:	Lecturer edits his timetable based on the courses he selected
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	Students timetables get updated based on the courses they selected

NR	14
Summary:	Notifications
Actor:	Lecturer
Description:	Lecturer gets notifications from university or from individual students or classes
Precondition:	Must be successfully authenticated
Alternative:	None
Postcondition:	None

APPENDIX

The appendixes are not always considered part of the actual Requirements Specification and are not always necessary. They may include

- Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
- Supporting or background information that can help the readers of the Requirements Specification;
- A description of the problems to be solved by the system;
- Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the Requirements Specification should explicitly state whether or not the appendixes are to be considered part of the requirements.

Appendix A. Definitions, Acronyms, and Abbreviations

Define all terms, acronyms, and abbreviations used in this document.

Appendix B. References

List all the documents and other materials referenced in this document.

Appendix C. Requirements Traceability Matrix

The following trace matrix examples show one possible use of naming standards for deliverables (FunctionalArea-DocType-NN). The number has no other meaning than to keep the documents unique. For example, the Bargaining Unit Assignment Process Flow would be BUA-PF-01.

For example (1):

Business Requirement	Area	Deliverables	Status
BR_LR_01 The system should validate the relationship between Bargaining Unit/Location and Job Class.---Comments: Business Process = "Assigning a Bargaining Unit to an Appointment" (Priority 1)	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted
		BUA-PF-01 Derive Bargaining Unit-Process Flow Diagram	Accepted
		BUA-PF-01 Derive Bargaining Unit-Process Flow Diagram	Accepted
BR_LR_09 The system should provide the capability for the Labor Relations Office to maintain the job class/union relationship.---Comments: Business Process = "Maintenance" (Priority 1)	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted
		BUA-PF-02 BU Assignment Rules Maint Process Flow Diagram	ReadyForReview

For example (2):

BizReqID	Pri	Major Area	DevTstItems DelivID	Deliv Name	Status
BR_LR_01	1	BUA	BUA-CD-01	Assign BU Conceptual Design	Accepted
BR_LR_01	1	BUA	BUA-DS-02	Bargaining Unit Assignment DB Modification Description	Accepted

BizReqID	Pri	Major Area	DevTstItems DelivID	Deliv Name	Status
BR_LR_01	1	BUA	BUA-PF-01	Derive Bargaining Unit-Process Flow Diagram	Accepted
BR_LR_01	1	BUA	BUA-UCD-01	BU Assign LR UseCase Diagram	ReadyForReview
BR_LR_01	1	BUA	BUA-UCT-001	BU Assignment by PC UseCase - Add Appointment and Derive UBU	Reviewed
BR_LR_01	1	BUA	BUA-UCT-002	BU Assignment by PC UseCase - Add Appointment (UBU Not Found)	Reviewed
BR_LR_01	1	BUA	BUA-UCT-006	BU Assignment by PC UseCase - Modify Appointment (Removed UBU)	Reviewed
BR_LR_09	1	BUA	BUA-CD-01	Assign BU Conceptual Design	Accepted
BR_LR_09	1	BUA	BUA-DS-02	Bargaining Unit Assignment DB Modification Description	Accepted
BR_LR_09	1	BUA	BUA-PF-02	BU Assignment Rules Maint Process Flow Diagram	Accepted
BR_LR_09	1	BUA	BUA-UCD-03	BU Assign Rules Maint UseCase Diagram	Reviewed
BR_LR_09	1	BUA	BUA-UCT-045	BU Assignment Rules Maint: Successfully Add New Assignment Rule	Reviewed
BR_LR_09	1	BUA	BUA-UCT-051	BU Assignment Rules MaintUseCase: Modify Rule	Reviewed
BR_LR_09	1	BUA	BUA-UCT-053	BU Assignment Rules MaintUseCase - Review Assignment Rules	Reviewed
BR_LR_09	1	BUA	BUA-UCT-057	BU Assignment Rules MaintUseCase: Inactivate Last Rule for a BU	Reviewed
BR_LR_09	1	BUA	BUA-UI-02	BU AssignRules Maint UI Mockups	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-021	BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Success	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-027	BU Assignment Rules Maint TestCase: Modify Rule - Success	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-035	BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Error Condition	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-049	BU Assignment Rules Maint TestCase: Modify Rule - Error Condition	ReadyForReview

For example (3):

BizReqID	CD01	CD02	CD03	CD04	UI01	UI02	UCT01	UCT02	UCT03	TC01	TC02	TC03	TC04
BR_LR_01			X		X		X			X		X	
BR_LR_09	X			X		X			X		X		X
BR_LR_10	X			X					X		X		
BR_LR_11		X											

Appendix D. Organizing the Requirements

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

By System Mode
Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

By User Class
Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

By Objects
Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

By Feature
A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

By Stimulus
Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

By Response
Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

By Functional Hierarchy
When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

Additional Comments
Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.