

CHAPTER THREE

The Requirements Set

Chapter Key Topics

- Requirement categories
- Requirement organization
- Organization benefits

The Internet Impact

The Internet opens doors for many new business opportunities. Each business community from one or multiple corporations is involved in producing a successful product through this medium. The implementation of an Internet solution involves capturing requirements from all the business communities. Organization of the requirements assists in identifying gaps and conflicts sooner in the process, when the cost of change is less.

This chapter introduces a *requirements set framework* that organizes individual requirements into specific categories. The purpose of this organization is to assist in the validation of requirement coverage for the problem domain, in this case, the Internet product. By arranging requirements in an organized fashion, gaps in knowledge and inconsistencies can be identified more readily and corrected earlier in the process. Organizing the requirements set according to the framework increases the speed with which products can be developed because its structure works with the evolutionary process and promotes parallelism.

While discussing the evolution of the requirements, different types of requirements emerged (see Figure 3.1). These are:

- **Initiating requirements:** requirements that clarify the concept as it pertains to the business objective. These are typically the requirements identified by the sponsor of the business idea in the business case.
- **Primary requirements:** business requirements that need to be satisfied in order to fulfill the concept. These are the requirements that each business community has identified as its scope in order to satisfy the business objective.
- **Derived requirements:** requirements that need to be satisfied in order to satisfy the primary requirements. Derived requirements come in two types:
 1. *Supporting:* requirements that have been allocated to a specific category because of their ability to supply essential details.
 2. *Association:* requirements that have been written to support other requirements in a different category of the requirements set.

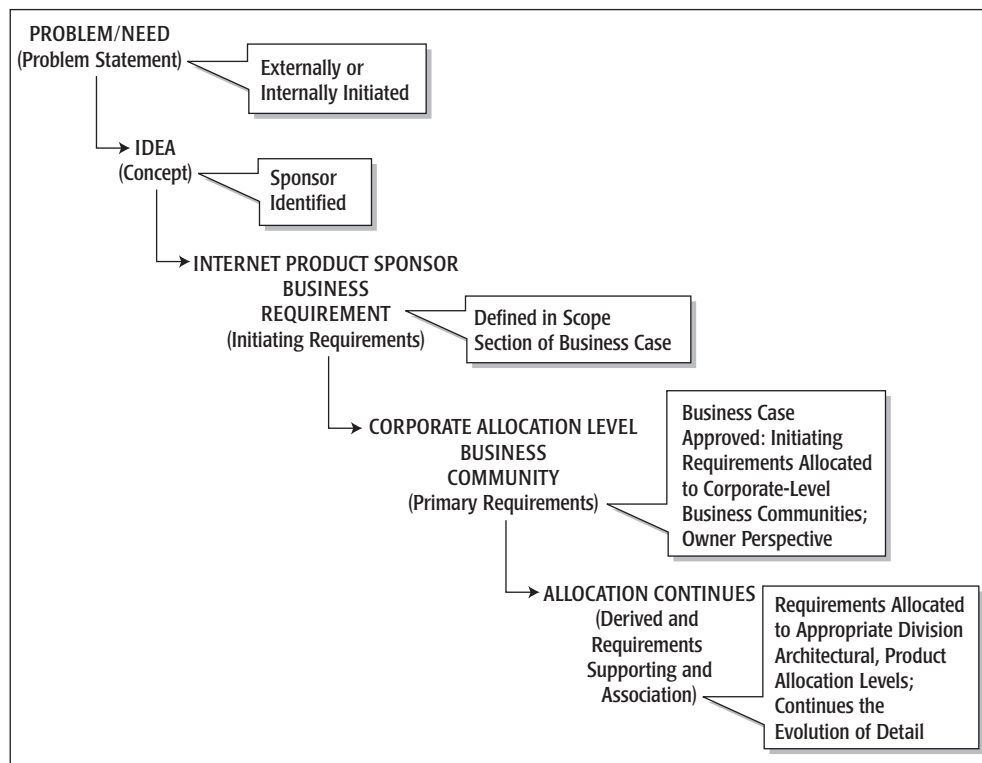


Figure 3.1 Requirement Types through Allocation

Requirement Categories

75

All of these types of requirements can be categorized and organized within the requirements set framework. All of them need the same components to create a high-quality requirement: identifier, description, priority, validation criteria, source, and so forth. The requirements set framework does not enforce the individual quality rules; it only categorizes the individual requirements of the requirements set into a specific organization. The category, and not the type of requirement, defines its placement in the requirements set framework.

Requirement Categories

A *requirement category* is the subject matter of the requirement. It takes three dimensions to properly categorize each individual requirement. These dimensions clarify the special interest of the subject matter and refer to the type of questions that need to be answered in order to define the specific needs for the category. These questions, when applied to the requirements set framework, create the requirements pattern.

The three dimensions are

1. *Community*: corporate function
2. *Perspective*: level of detail
3. *Focus*: specialized view

A specific requirement is categorized before placing it into the requirements set framework (see Figure 3.2). The figure identifies a fourth category, *association*. This additional category is required to meld, link, or associate the different subject matter requirements together. To avoid confusion, this fundamental discussion appears later in the chapter. For now, let's review the different dimensions in closer detail.

Requirement Community

During the requirements process, each requirement is assigned (allocated) to different organizational areas that develop the requirement in order to satisfy the business objective. These different business communities correlate to the allocation process discussed in the previous chapter. The business objective of an Internet product is to develop a B2B product that will have the scope delivered to each of the business organizations. Each organization will identify a list of primary requirements to be incorporated into its own specific deliverables.

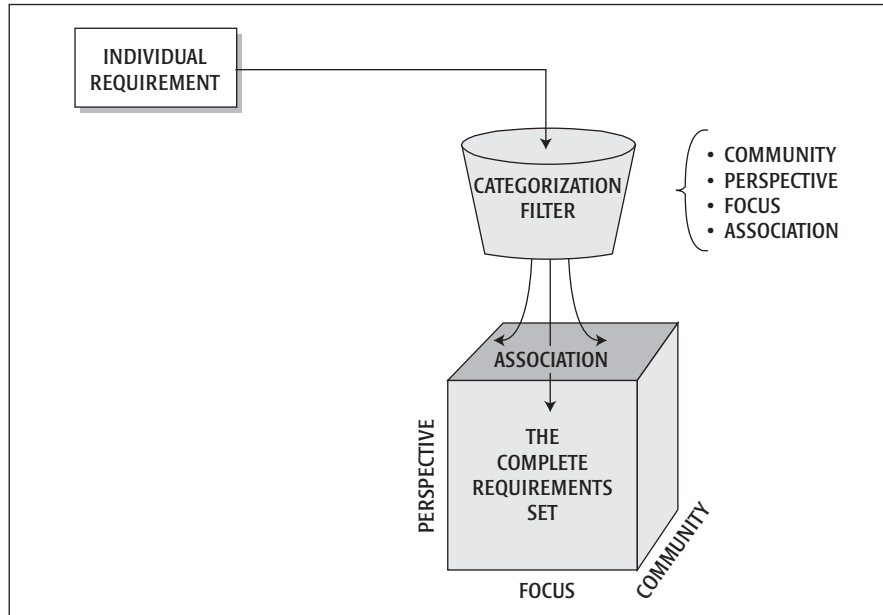


Figure 3.2 Requirement Categorization

The communities are separated into four business areas (see Figure 3.3). Each of these areas can be subdivided into finer organizational communities. The four main corporate or business community areas are:

1. *Business practice*: organizational divisions directly associated with the businesses that relate directly to the business objective. These include business areas such as product development, marketing, and customer service. For e-business, there may be multiple product objectives, one for each business in the B2B connection. In this scenario, a single goal for the product will have specific business objectives that correlate specifically to the company's purpose.

For example (see Figure 3.4), a book retailer has a B2B link with a shipping company. Each of these companies has its own product that is distinguished from the other company's product. Each has a common product objective of streamlining the product order to the logistics of shipment. Each company has a business project with different corporate objectives but the same product objective. Each of these companies will have to develop

Requirement Categories

77

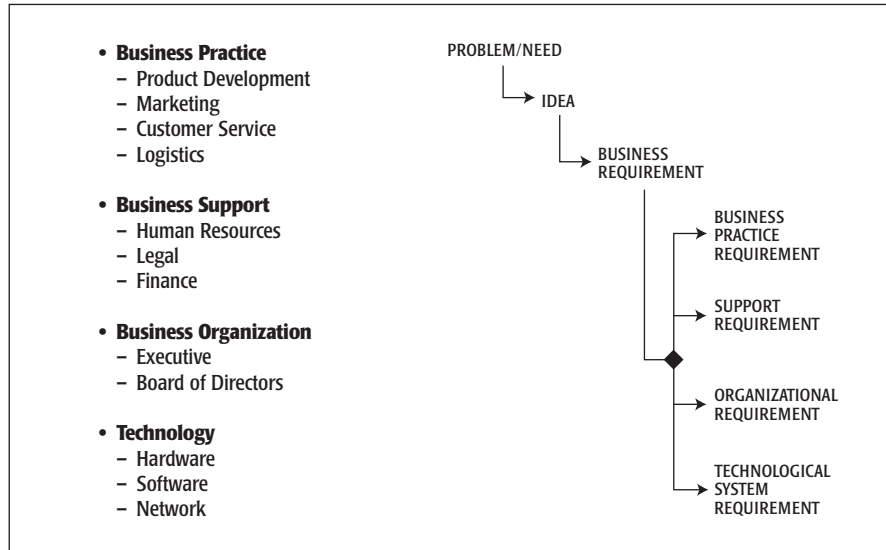


Figure 3.3 Requirement Community Category

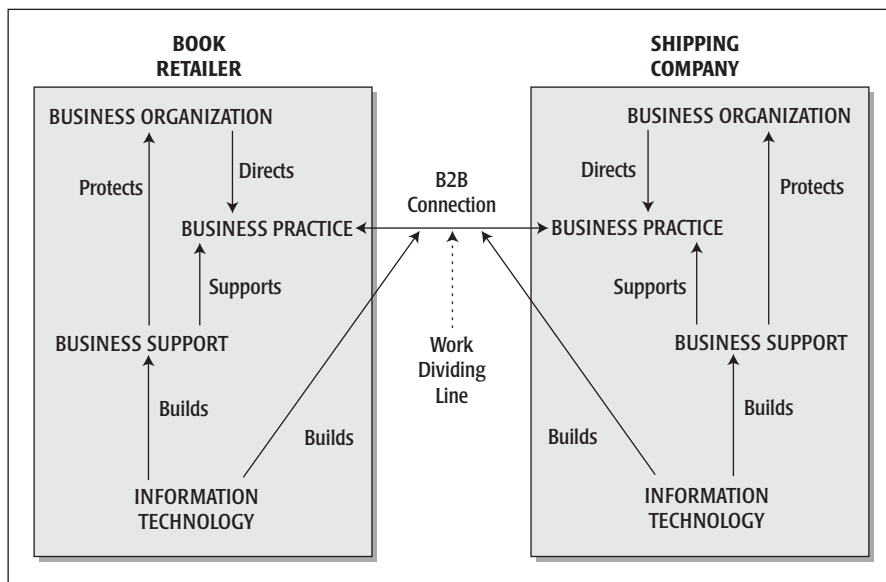


Figure 3.4 Business-to-Business (B2B) Community Category Relationships

and manage *business practice community requirements*. And, finally, each of the companies has its own requirements set. The links (association) between the requirements are additional requirements defined by both organizational communities. This will enable them to identify the impact of a change instituted by one corporation on the other.

2. *Business support*: organizational divisions that support the running of the organization. These include human resources, finance, and legal. Their objective is to support the project by identifying requirements that need satisfying to meet the Internet product objectives. In a B2B scenario, the business support areas support their own company's interest. No cross-company correlation is necessary.
3. *Business organization*: divisions that dictate direction and business objectives, including the executive committee and the board of directors. Their objective is to support the Internet project by identifying requirements that need satisfying to meet the product objectives. As with business support, in a B2B scenario, the business organization areas support their own company's interest. They are instrumental in developing the vision for forming the relationship between B2B companies. It may be necessary to correlate these requirements across corporate requirements sets if one company requirement is dependent on the other.
4. *Technology*: divisions that relate only to the technological aspects of supporting the business objective. This does not include systems that developed as products to be sold. For example, Microsoft applications are products developed under the community business practice. These are external technology products. The sales-tracking systems, however, fall under the technology community. These are internal technology products. In a B2B situation, they are not only working to connect with different systems within their corporate boundaries but also are responsible for working jointly with other companies to develop the connection to the B2B company. A requirement could be dependent upon another requirement being satisfied by another company. The technology community may also need to develop derived requirements that will support the other company's primary requirements.

Just to make things more complicated, each of the first three communities receives requests to build technology products (external or internal). In fact, so too will the technology area (for example, system software/hardware upgrades). Each request needs to satisfy the needs of one, two, three, or all four of the communities.

Requirement Categories

79

Here, then, lies the definition of community. *Community* segments the types of deliverables according to business organization. Each deliverable is specific to meeting the needs of the business community. The needs (that is, requirements) will continue to evolve and be allocated further into the community (corporate to division to architectural to product levels) to build the deliverable for the business community. The evolutionary state within the business community can be correlated to perspective, discussed next.

Requirement Perspective

Perspective relates to the evolution of the Internet product from idea to the implemented solution. Throughout the evolution of requirements, they will be further refined and specific details will be provided to enable the building of the solution. Through each step, a level of extraction takes place that eventually leads to the specific parts of the whole Internet product. A separate deliverable is produced for each perspective. Each deliverable takes a step toward building the product. Skipping an evolutionary step may provide a solution, but the solution will be flawed due to the missing detail or missing requirement. The solution will probably be delayed due to the need to go back and get the detail or requirement that should have been obtained earlier in the process. The final product is built but on incorrect information.

The idea phase (that is, *initiating requirement*) identifies the business community that has the primary interest in the product. The idea is then allocated to each community to develop the initial set of primary requirements (*scoping* and *business requirements*). Each community has its own perspective. Each perspective evolves the community's requirements through its specific allocation levels, developing essential details to the requirements and developing additional requirements (*derived requirements*) that are needed to support the idea. Different individuals are responsible for developing the requirements for the community and perspective. Each of these individuals has a specific point of view as to the categories of requirements that need to be captured. This specific point of view relates to the third requirements set category: requirement focus.

Requirement Focus

Focus looks at each perspective for the community from an abstract view of the product to be developed. For example, with respect to information systems, the focus would be the type of system component: interfaces, data, process, network,

timing, rules, and constraints. The role of a specific community is to be responsible for creating an end deliverable. Through the evolution of requirements, the role has a specific, tailored view of the overall business idea. The person responsible for that focus develops a slice of the product that evolves through the same perspectives for the community as any other view.

For data focus,¹ the list of entities evolves into an entity relationship. This high-level data model evolves into a fully attributed business model. This model is customized for the technical architecture which, in turn, is transformed into a data language that runs in production.

Relationships between Categories

To recap, *community* refers to the business unit, *perspective* relates to the level of abstract detail for the community, and *focus* pertains to the point of view of the community and perspective. The combination of community, perspective, and focus defines the *category*. The category is represented by a *cell* in the *framework*. Following a requirements process (and subprocess) fills the framework. To assist in filling the framework, the *requirements pattern* has questions that are tailored to the subject matter for each cell or category.

Requirement Organization

The development of any Internet-based application requires the development of different work products that stem from the idea. Each of these work products is needed to build a high-quality Internet-based application. Each of these individual work products follows a process similar to a manufacturing process described in the previous chapter. Community, perspective, and focus are dimensions of the organization, which when working together will develop one or more of these work products.

This process is no different from any other manufacturing process. The perspective is the evolution of any product. The development of any product requires different roles for building each piece of the product. To explain the

1. This is for illustrative purposes only. The requirements set framework holds requirements. How the requirements are captured, analyzed, and specified is independent of method or technique.

finer details of the framework and the association between categories, the example of building a house is used below.²

A Quality Home

When a house is being built, the idea (or architectural plan) evolves through perspectives (levels of abstraction from the concept) to produce a deliverable (the home). Each different perspective refines the details until the house is built. Each perspective is paired with a specific individual (or contractor) to build a deliverable that assists in creating the final product (see Table 3.1). The deliverable is built according to the perspective of the responsible individual. Each of these phases evolves from the information gathered and analyzed from the previous phase. Each phase has the perspective of another individual role. Figure 3.5 shows those involved in developing a home. Each individual with his own focus evolves deliverables (work products) through the different perspectives.

Skipping a perspective in building the home increases the risk of building a faulty home. No one would want to live in a home they knew had skipped a

Table 3.1 Manufacturing a Home

Manufacturing Phase	Architectural Output	Perspective
The birth of an idea	A ballpark view of how big, how many rooms, etc.; documents the scope	Planner
Analyzing the feasibility	A small model depicting and confirming understanding; provides the vision and funding	Owner
Designing the product	A detailed design of what is to be built; specifies the physical design	Designer
Developing the product	The specifications of how to build it; constructs the physical product	Builder
Assembling and testing the product	The individual components; provides the components	Subcontractor
Implementing the product	The house; utilizes the end product	User

2. For the purposes of this discussion, only the phases up to the first implementation are discussed. Each iteration of an Internet-based application evolves similarly.



Figure 3.5 Perspectives and Focus Areas for Building a Quality Home

phase of this process. Instead, inspectors are available at each phase (perspective) to spot possible defects and correct them early in the process (quality gate). The cost of correction after implementation, if possible at all, is too risky and expensive. To further reduce the cost of building a home, specifications or any of the deliverables from each phase can be reused or expanded upon to build other homes.³ This is only doable if the deliverable passed quality inspections for the original product.

The House That Zachman Built

In 1987 while working for IBM, John A. Zachman published a framework for discussing information systems architecture.⁴ He described the analogy between the software development process and the construction and manufacturing processes. This architecture is the basis of the requirements set framework. To understand the requirements set framework, it is important to understand John Zachman's architecture.

3. One of the unique characteristics of Former President James Carter's Houses for Humanity program (under Habitat for Humanity) is the reuse of standard plans. Each house is built from the same plan following the same process. Each reuse of the specification reduces the cost without sacrificing the quality of each house.

4. See Zachman 1987.

Zachman illustrated how the process, deliverables, perspective, and focus are conceptually the same. The process can also conceptually be the same for different business communities (for example, legal, human resources, and shipping). For simplicity, we will focus on software products for this example. The output of software development deliverables correlates to architectural output (see Table 3.2). The table documents the evolution of the software detail and how it is a process similar to construction and manufacturing.

Different Perspectives for Either Process

The roles involved in building a home (see Table 3.3) identify the need for different perspectives. Each individual has a responsibility to concentrate on his particular perspective to ensure the building of a quality end product. As the evolution of the idea to final product occurs, different phases are followed through the different perspectives. Perspectives reflect the areas of responsibility of each role. For example, a designer's perspective is quite different from the owner's perspective;

Table 3.2 Software Development Deliverables Compared with Architectural Output

Manufacturing Phase	Architectural Output	Information Technology Output
The birth of an idea	A ballpark view of how big, how many rooms, etc.; documents the scope	The scope of the investigation for the software product
Analyzing the feasibility	A small model depicting and confirming understanding; provides the vision and funding	The vision or business view of the software product to be developed along with the appropriate funding
Designing the product	A detailed physical design of what is to be built	The detailed physical representation of the software product to be built
Developing the product	The specifications of how to build it; constructs the physical product	The detail specifications of how the software is to be built
Assembling and testing the product	Provides the individual components	The actual software code and file definitions
Implementing the product	The house; utilizes the end product	The software product

Table 3.3 Information Technology Community Perspective

Manufacturing Phase	The Architectural Perspective	The Information Technology Perspective^a
The birth of an idea	Planner	Business executive
Analyzing the feasibility	Owner	Business representative ^b
Designing the product	Designer	Systems analyst
Developing the product	Builder	Developer/programmer
Assembling and testing the product	Subcontractor	Quality analyst
Implementing the product	User	Business end user

a. The names for these roles vary from organization to organization. For the purposes of this discussion, the most commonly used role titles are used. These roles are not all the roles that are involved in developing or managing the deliverable. Multiple roles may be performed by the same individual in small organizations.

b. This phase is accomplished with the assistance of a requirements engineer who can also assist at any of these phases.

the builder's perspective is different from a designer's.⁵ What is important is that the owner's perspective is satisfied in the final product. Though each role has a different perspective, all roles work together as a team to build the quality end product.⁶

The same is true with software development (including Internet-based products). Each phase (see Table 3.3) has a different perspective. Each phase develops another level of detail to build the final quality product. What is important is that the perspective of the "owner" (business executive) is satisfied in the final product.

Different Perspectives for Requirements

Each of the Zachman framework perspectives involved defining requirements and requirement details (see Table 3.4). The first three perspectives are requirements that are more commonly referred to as "business requirements." The first perspective (planner) identifies the list of what is in and out of scope (primary

5. Material derived from a demonstration of Structure for Information Systems Architecture by Framework Software, Inc., www.frameworksoft.com.

6. Material derived from Bruce 1992.

Table 3.4 Perspective Requirements

Manufacturing Phase	Zachman Perspective	Information Technology Translation	Requirement Deliverable
The birth of an idea	Entity classes and scope (<i>planner</i>)	Defines the scope of the investigation	Lists descriptive needs by each business community that define the scope of the investigation; includes lists of what is to be investigated and what is not included
Analyzing the feasibility	Model of the business (<i>owner</i>)	Defines how the business communities view the scope	Takes the lists one step into descriptive detail by providing association and dependencies between requirements (relationships)
Designing the product	Model of the information system (<i>designer</i>)	Defines the essential details to run the business; still technology independent	Each requirement and its relationship with other requirements within and outside of the business community fully attributed with the details that clearly define the need
Developing the product	Technology model (<i>builder</i>)	Defines the technological solution to the logical view; technology dependent	Derived requirements associated to specific needs of the builders to help them develop the solution to the requirements
Assembling and testing the product	Technology model (<i>subcontractor</i>)	Assembly and testing	Derived requirements associated to specific needs of the testers and distributors of the solution to help them implement the solution to the requirements
Implementing the product	Functioning system (<i>user</i>)	Implementation	Continue capturing feedback for process improvement or product improvement requirements

requirements) as it relates to the initiating requirement (the product concept). The second perspective (owner) defines the requirements and their dependencies (derived requirements) on other requirements. The third perspective (designer) defines the specific details that are essential in defining the requirements and their relationship with other requirements. The details may include additional derived requirements due to the specific interaction between business communities.

The fourth perspective (builder) is the break away from the business requirements. These define specific requirements (derived requirements) that need to be satisfied in order to build the solution. These requirements may or may not be captured by the requirements engineer. In some companies, technology specialists such as a database administrator or technical architect may capture these requirements. In truth, these specialists are the suppliers of this perspective's requirements. It is the responsibility of the requirements engineer to ensure that the requirements are elicited, analyzed, specified, validated, and approved (especially high-price items or items that go against corporate standards) and to ensure they are managed.

The fifth perspective (subcontractor) begins the process for the next iteration of the product. These requirements include what metrics or information should be captured in order to judge the success of the product. For Internet applications, requirements may state the need to capture customer feedback, click history, response time, and sales figures. These are requirements geared toward understanding the success of the first iteration and providing valuable information that could foster a change in direction to retain market presence.

The sixth perspective (user) is the implemented product or iteration. Requirements relate more to the next iteration of the product, thus initiating the next cycle, which begins with the planner perspective.

Each business community has the same perspectives; each perspective for each community has a different specification. Each perspective follows the same subprocess to elicit, analyze, specify, validate, and approve the requirements for its community and perspective.

Different Views of the Same House

When building a home, each phase or perspective requires a different focus or view. These views concentrate on a specific area for that perspective. For example, the house requires an architect's view, a plumber's view, an electrician's view, a landscaper's view, and so forth (see Table 3.5). Each has similar deliverables: scope, owner's model, physical model, specifications, components, and the final product. Each deliverable at each perspective level has its own focus. Each focus (see Table 3.5) still has the same architectural perspective. Nothing can be viewed independently! All views and perspectives are important to build a high-quality house. Each focus area must narrow down a concept into finite detail to ensure compliance and quality and to ultimately meet the owner's needs.

Table 3.5 Focuses for Building a Home

The Architectural Perspective	Architect's Focus	Plumber's Focus	Electrician's Focus	Landscaper's Focus
<i>Planner:</i> documents scope	<i>Planner:</i> number of rooms, size of property, etc.	<i>Planner:</i> number of facilities, features, etc.	<i>Planner:</i> number of connections, etc.	<i>Planner:</i> number of shrubs, trees, sprinklers, etc.
<i>Owner:</i> documents view	<i>Owner:</i> look of building	<i>Owner:</i> look and location of kitchens and bathrooms	<i>Owner:</i> each room's electrical requirements	<i>Owner:</i> layout of the outside
<i>Designer:</i> documents design	<i>Designer:</i> physical design and material of the house	<i>Designer:</i> physical design and material for the water rooms	<i>Designer:</i> physical design and material of the electrical system	<i>Designer:</i> physical design and material of the landscape
<i>Builder:</i> documents specifications	<i>Builder:</i> detailed specifications on how to build the house	<i>Builder:</i> detailed specifications on how to insert the plumbing	<i>Builder:</i> detailed specifications on how to wire the house	<i>Builder:</i> detailed specifications on what and where to plant
<i>Subcontractor:</i> documents components	<i>Subcontractor:</i> the building	<i>Subcontractor:</i> the plumbing	<i>Subcontractor:</i> the electrical wiring	<i>Subcontractor:</i> the trees and shrubs
<i>User:</i> uses the final product	<i>User:</i> the home	<i>User:</i> the kitchen, laundry, and bathrooms	<i>User:</i> the electrical outlets and fixtures	<i>User:</i> the landscape

Different Focuses of the Same Software Solution

Building software solutions also requires different focuses. In his Information Systems Architecture, John Zachman identified the following areas of focus:⁷

- **Who** is involved (people/organization/systems)?
- **What** is it made of (data)?
- **Where** are things located (network)?
- **When** do things happen (business events)?
- **Why** are things done (business policy)?
- **How** does it function (process)?

7. "A Framework for Information Systems Architecture." John A. Zachman. IBM Systems Journal, Vol. 26, No. 3, 1987. IBM Publication G321-5298 (www.zifa.com).

Though we are concentrating on the business area of information technology, the same focuses apply to each of the other business communities: business practice, business organization, and business support. Each of the business communities has a deliverable that has “who, what, where, when, why, and how” requirements to capture. They all need to be elicited, analyzed, specified, validated, and approved in order to build the specific deliverable.

The purpose of drawings, models, and other documentation deliverables is to enable the reviewers and those who must act on the deliverables to relate to them and to agree or disagree with the contents. The deliverables, in respect to requirements, are specifications used to validate for quality and to build upon.

Let’s walk through the focus (special interest) areas to obtain an understanding of the specifics of each one. Note that each of these focus areas captures the *functionality* the product must have. This functionality lives with the product through each iteration until either the requirement is no longer needed or the entire product is no longer needed.

The People (“Who”) View

This view focuses on associations to the product. The importance is that all the people, departments, organizations, and systems that interface with the product are well documented as to their needs. They trigger action on the part of the product.

The “who” needs include supplying information, accessing information, triggering functionality, and receiving results (such as reports). For example, a customer may submit an order and receive merchandise. The Internal Revenue Service may also be involved by being the recipient of quarterly financial reports. For the book retailer example, the shipping company personnel may have access to the book retailer’s system in addition to the book retailer personnel having access to portions of the shipping system. For B2B, both forms of access from opposite systems are documented. In the world of the Internet, businesses, including competitors, are users. This category of requirements may need information such as supplier data.

It is important to also note who should *not* have access to the product. This observation spawns additional security-related requirements. The most common example in traditional manufacturing is the childproof cap used on medicine bottles. For the book retailer application, requirements could call for a user-initiated security key of the user’s specified functionality and information. The objective is to prevent hackers from obtaining valuable customer or company information.

“Who” refers to the product interfaces. At each perspective, the interfaces may evolve into identifying security requirements as to what information they may have access to and what functionality they may perform. At the first perspective, you have a list of who should have access. You have a second list that clearly defines who should not have access. The “should not” list evolves as you refine the requirements through each perspective for specific functionality and information.

The Information (‘What’) View

This view focuses on the information requirements of the product. For software, this describes all the data needed to fulfill the requirement. For the book retailer and shipping connection, examples are the pickup location (warehouse) and the delivery location (customer site).

“What” represents the data needed by the Internet-based application. Data come in different forms, thanks to multimedia. Data can be textual, video, graphical, animation, and audio, all of which can be transmitted via files or messages across the Internet on cable, telephone, or wireless connections. The information can be distributed to PCs, laptops, televisions, personal digital assistants, and cellular telephones. All of these details are captured as requirements evolve through the different perspectives.

Data or object models are the most common deliverables in defining the “what” requirements. The models are more than just diagrams. A model contains textual information about the contents of the diagram. In fact, the diagram *supports* the textual description. Data and entity classes can be classified as information models for the purpose of explaining the “what” focus area.

An information model defines the common business vocabulary. Getting concurrence on specific definitions is the challenge of all good modelers. The models and the diagram depict relationships. These relationships define many of the business rules that must be supported. For example, a person cannot place an order without supplying at least one valid credit card number and a valid address. In the United States, the credit card address may be a post office box, but a shipping address must be a street address (a common oversight made by many online shopping sites) if a non–United States Postal Service (USPS) company is used.

Technical information models organize the information for performance for a specific technology. They are a step removed from defining the business. Both the business information model (defined at early perspective levels) and the technical information model are required. The technical model is built *from* the business model and will change more frequently than the business model. These changes are typically intended to improve performance or to take advantage of

new technologies. These technology-related changes rarely affect the business model. However, any changes to the business model must be reflected in the technology model. The business model defines the business; the technology model defines how the business is supported technically. The businesspeople validate the business model, whereas technicians (and requirements engineers) validate the technology model.

In many business information models, audit trails may be left to the technology models. This is not true in the Internet world. When capturing “what” requirements, it is important to include confidential information that will provide audit trails. This requires information about the data that was captured: when and by whom. It must be proven that a transaction took place (especially changes and deletions). This information must be incorporated and approved by the business community.

The Location (“Where”) View

“Where,” the third focus area, represents the location or networking type of requirement. The “where” requirements, as with the “who” and “what” requirements, evolve through the different perspectives. Unfortunately, many requirements engineers capture this type of information later rather than earlier in the process.

Network engineers must meet the expectations of the different users as well as maintain the network infrastructure for the organization. Their requirement needs may be captured in other focus areas, but they must still take that information and reformulate it into their own models, evolving them into router configurations and other network work products. The information they need to capture includes

- *The type of access by user, event, and function.* The type of access could be batch file, online access, wireless access, or Internet access. Knowing this information assists in determining the type of communication that is needed to meet the requirement of the user.
- *What is needed at each location.* This includes what data and how much data will be transmitted from one node to another. How often will the transactions be executed? Are there peak periods? The objective is to determine the load on the network.
- *Security requirements for each location.* This includes security at the data and function by actor (“who”) and access type. For example, a wireless transmission to a mobile telephone may be restricted to only summary information.

The “where” view focuses on connectivity. This entails describing the desired location where the product will be used and who must have access to it. For a stock-trading system, locations may be worldwide. Wireless has added a new dimension to the Internet. Devices such as handheld organizers can connect to the Internet application from anywhere. No longer must a device be physically attached. For the “where” requirements of an Internet application, location pertains to the traffic concentration areas for the customer base and the functions and amount of data they transmit.

The Event (“When”) View

This view focuses on timings that trigger action by the product. It identifies when things must or usually occur, for example, the closing of the U.S. stock market at 4 P.M. EST. Can the book retailer accept the order for a next-day delivery request with the shipping company’s logistics plan? The shipping company may have rules on what they can pick up late in the day and still be able to deliver to certain locations within a 24-hour period.

There are four basic types of events:

1. Initiated by time (for example, order received past shipper’s latest pickup time)
2. Initiated by action from outside the product (for example, order placed for next-day shipment)
3. Conditionally triggered (for example, back-order products arrived at warehouse)
4. Arrival of information (for example, order picked up)

There are different models that can be created to reflect “when” requirements. The first perspective includes lists of the events with a list of associated responses. The events are to be correlated (requirement relationship) with “who” requirements in the detailed perspective, along with a list of conditions and actions the event incurs. When capturing the essential requirement details, the correlations between “what,” and “how,” and “where” as well as other focus areas have not yet been discussed. The point is that just as with any of the other focus areas, the “when” requirements evolve in detail and level of abstraction through the different perspectives.

The Rule (“Why”) View

This view focuses on business rules and policies that must be adhered to when arriving at the solution. Rules are critical to the survival of a corporation. They

can *never* be violated. Rules are defined for both normal and abnormal business circumstances. For example, an automobile can be sold as “new” only once during its life. A stock trader cannot cancel a submitted stock trade (buy) if the other party has already submitted his side (sell) of the transaction. The information about a customer order cannot be displayed if it is marked as secure without a password being supplied by the user.

The Internet has changed the way business is conducted. This means that the current business rules and policies that exist may need to be changed or tweaked or perhaps totally abandoned. This is a new economy based on a new business model that is filled with a new set of business rules. *Keeping outdated business rules in place can limit business opportunities on the Net!*⁸

Rules define the policies for a particular situation. When the situation in question occurs, it is an obligation that the rule be satisfied. These rules may affect the other focus areas, and these other focus area models must include a correlation to the business rule.

Some rules are defined outside the organization, including governmental (legal, IRS) and others rules dictated by competitors or the culture of the times. For example, state laws may prohibit a car from being sold as new when the mileage exceeds a specific number, which varies from state to state. Another example is the tax applied to the purchase. Is the ship-to or the ship-from tax charged according to the product's or service's value, and is it according to the date of order or the date shipped? This also varies from state to state and from country to country.

Policies that control the processing of the organization are usually defined internally. They define the culture and style of the organization. For example, the business rule for an Internet application for purchasing merchandise may state that the item must be requested from the supplier when the inventory dips below a specific level. Each item in the inventory may have its own specific reorder business rule.

Rules can also be technical in nature. For example, when the transaction reaches a specific level on the network, additional lines, typically reserved for backup situations, will be used to handle short periods of unexpected traffic.

8. Automobile manufacturers keep cars at individual dealers' car lots. For Internet sales, why should a dealer offer up one of its cars at a reduced cost? Why should the manufacturer have to pay dealer commission if a deal was not involved? New business rules concerning dealer commission had to be redefined to support the new Internet medium for car sales.

There are five different types of business rules:

1. **Constraints:** *limiting instructions.* A person must respond to an end-of-auction notice within three days.
2. **Heuristics:** *characterization instructions.* A customer in good credit standing should be notified of sales of similar products that have been purchased by the customer in the past.
3. **Computations:** *calculation instructions.* Foreign currency will be calculated by using the value of the currency for the day the item was shipped multiplied by the price of the item.
4. **Inference:** *implied instructions.* A customer who charges 35 percent more than his average daily transaction level will be placed on the customer watch list.
5. **Timing:** *conditional instructions.* Once a back-ordered item is received in the warehouse, it must be shipped to the next customer in line by the end of the next business day.

Timing is a type of rule associated with or often documented as a “when” requirement. In fact, all the business rules must be associated with other focus-type requirements. They may correlate to “what, how, when, and where” requirements. They may affect community requirements other than those of information technology. It is important during validation to evaluate each business rule for its impact on other requirements.

The Process (“How”) View

This view focuses on the workflow of a specific function: the procedure, step by step. For example, an invoice that is 30 days past due requires a secondary notice and placement of the client’s name on the delinquent watch list. Alternatively, for the online access to your customized view, one of the process requirements could describe:

1. If an incorrect password is supplied to retrieve secure information, the user will be allowed three opportunities.
2. If after the third opportunity the correct password is not supplied, the secure data will not be displayed.
3. If the user has forgotten his password, he is allowed to select the “secret word” to remind himself of the password.

4. If the user assesses the “secret word,” another opportunity to supply a correct password is made available.
5. If the password is correct, the system displays the information.
6. If the password is incorrect, the system notifies the user that he can request a new password.

In our Internet example, the key business processes for the book retailer are

- *Order entry*: includes presentation and ease of use for the client
- *Payment*: involves calculating the cost of the product plus any surcharges (taxes and delivery) in the client currency via the acceptable methods (credit card, bank note, and so on)
- *Delivery*: involves packaging the requested item (virtual or physical) and delivering the goods after the payment has been determined acceptable (but not cleared)
- *Inventory management*: involves the supply chain
- *Settlement*: involves the actual conversion for goods and services

Volumetrics

The following rules of thumb depict the average number of functional requirements. This is helpful when you are planning the requirement effort. It is also helpful when taking a first cut at validation to identify whether you have a complete set of requirements for a specific community, perspective, and focus. Again, this is just a rough estimate to raise a potential flag that something may be missing from the requirements set.

- There should be a “how” requirement associated with every “when” requirement.
- There should be five “how” transactions requirements (including object methods) that impact a business entity “what” to support: (1) a create function, (2) a read function, (3) an update function, (4) a delete function, and (5) a list (includes search) function.
- There should be at least one requirement for each cell (category described by community, perspective, and focus).
- There should be a business rule “why” for every relationship between “what” entities or entity classes.
- There should be a “who” associated with every “how,” “when,” and “where” requirement.

These guidelines pertain to functional requirements. As shown in the next section, functional requirements are only a portion of the requirements set.

Extensions to the Information Systems Architecture

The Zachman framework for Information Systems Architecture ties the perspective, focus, and deliverable together for developing software (see Figure 3.6). To develop a business solution using information technology requires different *integrated* views and details. The contents of each cell identify possible deliverables for a software development project. This comprises the *functional* focus. These areas describe what features the Internet product must have and perform.

Additional Focuses

Two additional focuses are required to cover the needed information to build the product, as they constrain the product development decisions. One area describes

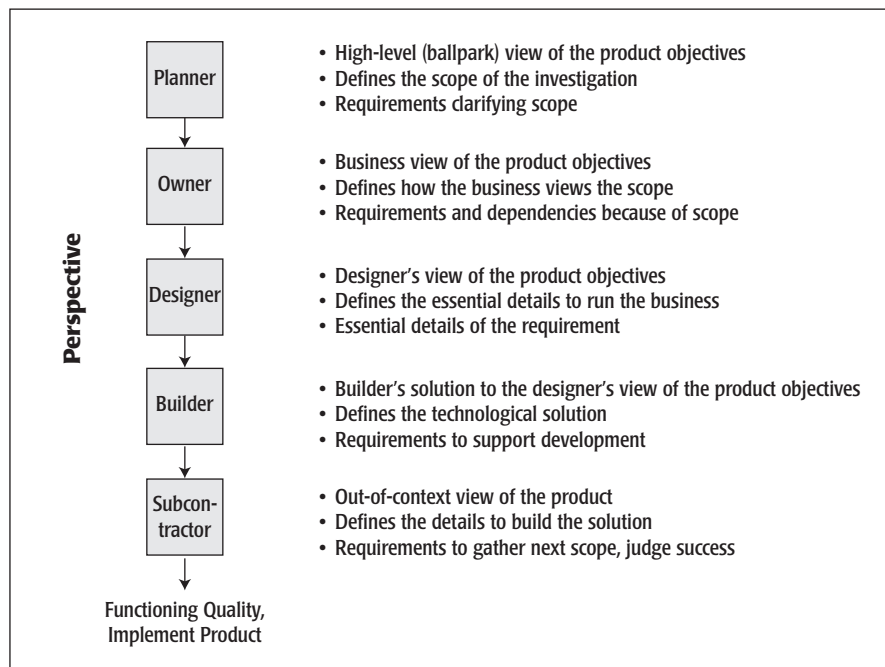


Figure 3.6 Perspectives Objectives

the constraints that must be applied to the product, and the other the constraints on running the project.⁹ These are called *nonfunctional requirements*.

1. **The product constraint's view** documents any restrictions that may affect how the product will be designed. In system engineering terms, these would include the system requirements¹⁰ pertaining to expectations as far as response time, security, and fault tolerance. Design constraints could also include restrictions as to the size of the application. Here they describe the environment and operation of the product. They may be referred to as quality of service (QoS) requirements.
2. **The project constraint's view** documents any restrictions that may affect the running of the project. This might include the number of resources and time constraints.¹¹ (Restrictions of tools, standards, and formats also fall under the category of project constraints.¹²)

The requirements that fall into these views should be written in a format that meets the same quality characteristics as functional requirements. They

9. Discussions continue in the field of requirements engineering as to whether constraints are requirements. In this book I chose to separate them from functional requirements but keep them within the list of focus areas. Constraints are needed for developing the product and could impact functional requirements. They should be elicited, analyzed, specified, verified, and managed.

10. The IEEE has a System Requirements Specification Template (Std 1233) implying that the system requirements are to be defined prior to software requirements. This is true as system requirements are identified, then allocated to hardware and software. Refer to Chapter Two under the discussion of requirement allocation.

11. The IEEE Software Requirements Specification Template (Std 830) is very clear in noting that a deadline is *not* a software requirement. It is a project constraint. Care must be taken to validate an imposed deliverable schedule. The deliverable date should be based on the approved functional, support, and design requirements. Delivery date should be negotiated during the review of the requirements. *However, there is a point in the life of the product where the revenue will decrease.* For example, a product, in order to meet projected revenue, must be the first to market or close thereafter. If the product does not meet that date, it might not be worth the time and effort for the company to build the product. This is an extremely important point! In reality, a delay in product delivery could cost the company a slice of global revenues or key holiday opportunities (for example, Mother's Day). Deadlines should be established but reviewed as to the feasibility during the validation phase. Negotiations will adjust the scope of the implementation effort to meet any market demand. Deadlines are requirements, just not software requirements!

12. Different companies developed parts of the Mars satellite. A missing project constraint was that all formulas were to use the same metric form of measurement.

Product Constraints

97

evolve through the same perspectives as the product's functional requirements. These additional focus areas include requirements that are crucial to the success of the Internet application.

Product Constraints

Product constraints, in the information technology world, refer to the “ilities.”¹³ To provide an example of different product constraints, the following six “ilities” are discussed in more detail below.

- Adaptability
- Reliability
- Scalability
- Security
- Usability
- Maintainability

The proceeding examples are crucial to the success of the Internet application.

Adaptability Requirements

One business model that is evolving is sometimes referred to as the “choice-board.” This is where the consumer, through the Internet, clicks on the features and prices of what he or she wishes to purchase. The manufacturer provides an online configurator that allows the consumer to select what options he or she is willing to buy and at what price. This new business model requires additional requirements that extend beyond software. This change also affects both the sales and manufacturing processes for the company.

Hence, the Internet application involves changes in the normal business process. In order for the product to succeed, changes in other business areas, or business communities, are required. The community represents the other business units that are impacted by the Internet application.

13. For a more complete list of nonfunctional requirements, see the IEEE System Requirement Specification (Std 1233). Remember that nonfunctional or QoS requirements pertain to product limitations. They constrain the designing and building of the product.



Table 3.7 Reliability (Performance) Requirements

	Performance 1				Performance 2				Performance 3			
Function	LOC. 1	LOC. 2	LOC. 3	LOC. 4	LOC. 1	LOC. 2	LOC. 3	LOC. 4	LOC. 1	LOC. 2	LOC. 3	LOC. 4
Function One												
Function Two												
Function Three												

ity for that location on their biggest trading day, equipment may be diverted to support the one-day peak instead of having excess capacity for the entire year. In any case, the requirements need to be explicit so those in charge of planning the capacity for hardware and networks have sufficient knowledge of the users' expectations.

Scalability Requirements

A good example of the importance of scalability was recently depicted in an advertisement in which a small group of entrepreneurially spirited individuals are standing around a personal computer. They are watching the opening of their new business venture, holding their breath as they eagerly await the first customer's order. Elated screams of success grow as the customer orders grow beyond their dreams. Then, quickly, the happiness turns to panic as the orders keep growing and growing. How are they going to satisfy what is now an unanticipated order volume?

This commercial illustrates the need for scalability on two fronts: technology and business community. Technology is the infrastructure that supports the Internet application. Business community is the business infrastructure that supports the product claims.

First, the technology. The technology must be scalable to adapt to the growth of the Internet application without the continual need to throw everything away and start over, the cost of which would deplete any revenue gain. Scalability is building an infrastructure that supports both vertical growth (increase in the number of sales for a specific product) as well as horizontal growth (increase in breadth of product).

It is important to anticipate some growth. Growth will impact the performance of the Internet application. The infrastructure must be flexible enough to support growth in network size, transaction handling, data throughput, page load timings, and security. This requires sophisticated tools that can model capacity planning on the site and network.

With regard to scalability technology requirements, it is important to document the potential growth in increments. A spreadsheet (see Table 3.8) or some CASE tools can be used to document these associative requirement details.

What is important is to determine the minimum, average, and maximum number of occurrences for each requirement. In other words, it is important to determine the anticipated volume of events (when), the number of transactions (how), and the number of users (by each type/actor). You also need to capture the timeframe for these volumes. For example, some time-initiated events may be dormant except when the time is triggered (close of the stock exchange or the end of the business day for that country). It is important for the network engineers to model the performance under the worst-case scenarios.

Growth will also impact the business infrastructure that is put in place. It is important to anticipate the risk of a supplier not being able to meet the growth of your business. Some small companies went out of business when UPS went on strike! eCost took a beating when 3Com could not get the chips for Palm Pilots. Scalability, in a business sense, requires risk analysis of all dependent business partners.

B2B supply-chain applications involve multiple vendors using multiple vendors. Scalability business requirements need to anticipate the possibility of a supplier not being able to meet the necessary volumes. In most cases, you can split the anticipated volume across multiple vendors. The business has minimal control over what vendors your vendors use. The scalability business requirements must contain scenarios of supply-chain interruption.

First, identify the possible breaks in the supply chain. Anticipate the worst scenario as well as the probable scenario. Table 3.8 can be adjusted to determine the severity of each possible outage.

At the scope level, scalability will be defined as *the need to implement a flexible infrastructure to support the projected growth*. At this point, the requirement is still ambiguous; all the volumetrics will most likely not be known. This requirement evolves as more requirements are captured. It is still important to document the scalability need at this point as a reminder that more information is required.

At the planner level, volumetrics begin to emerge. The business units should be able to provide projected numbers of the customers, orders, transactions, and so forth. These volumetrics should be applied to all models. Relationships among functional requirements are also being formed at the business level. As part of defining the relationships, the business unit should be able to project volumetrics about the relationships. For example:

1. Each potential client will search for five product descriptions, including price and availability, at one time.
2. One out of three searching clients will, on average, order two products with each order placed.
3. One out of five customers will bypass the search and place an order for one item.
4. The bulk of orders are placed between 10 A.M. and 1 P.M. EST on Mondays. The exception is the seven weeks before the end of the calendar year. During that time, the peak period expands to all weekdays.

Large companies that are dabbling in the Internet must not take these volume estimates lightly. There is a possibility that the Internet-based product may outperform the corporation's standard business, making the Internet the leading business unit.

Security Requirements

All of these business processes and information require some level of security processing. One type of security is required to protect the client, another to protect the corporation. The requirements engineer must spend time with all requirement suppliers on the impact of poor security. Security requirements must be captured on two fronts: (1) to protect the Internet user from intruders, and (2) to protect the corporation from theft or corruption (viruses).

It is critical to answer the simple question of what would be the impact to the corporation or the customer if data or processing were lost due to a security breach. If possible, attempt to obtain a quantifiable number at the following levels:

- Five-minute outage
- Day-long outage
- Week-long outage

Then try to determine through awareness-type questioning whether the data would actually be lost or corrupted. Sample scenarios include

- If data became corrupted
- If a competitor obtained any data
- If any file (data, voice, media) became corrupted

Clients must feel comfortable doing business with you. They must feel that you are reputable and that the payment process is a *private* transaction used for this purchase alone.

There are thieves among us. The first one who comes to mind is the predator who tries to destroy or capture corporate information.¹⁴ The other type of thief is one who wants to steal the product or service. When a thief of this sort obtains confidential information by using stolen payment options, it can cripple a company. Security must be taken very seriously and viewed as having a potentially negative impact at any level.

The seven worst security mistakes that senior executives make¹⁵ include four that can be documented during the requirements process:

1. Pretending the problem will go away if they ignore it
2. Relying primarily on a firewall
3. Failing to understand the relationship of information security to the business problem (understanding physical security but not seeing the consequences of poor information security)
4. Failing to realize how much money their information and organizational reputations are worth

It is up to the requirements engineer to capture security requirements and associate the need to “who,” “what,” “how,” and “where” as well as any interactions between these functional focus areas.

Usability Requirements

The Internet has evolved current business models to a point where the consumer holds the power. This has changed—and will continue to change—the way companies relate to their customers and compete with one another. Usability is a

14. This is discussed in more detail in Chapter Five.

15. The SANS Institute, *Newsletter*, May 2000, p. 7 (www.sans.org).

product constraint that directly speaks to the customer's ability to use the Internet product, and it must be captured. It should include

- The maximum allowed response time
- The availability to access 24 hours a day, seven days per week
- The ability to customize views at specific points
- The ability to filter information, minimizing what the user would perceive as "clutter data"
- The availability of choices for the product

Most importantly, a consideration for usability is how many clicks it takes to retrieve information and how many clicks to complete a workflow. Response time can be eaten away with every screen of updated or new information. This translates to time that the user is not willing to give up if he perceives that the usability of the application is at fault. The requirements engineer should capture this type of need and associate the information with each "how" requirement, similar to how you related scalability requirements. This assists the designers (users of these requirements) in developing an effective product with an emphasis on usability.

Maintainability Requirements

Every one to three months (sometimes sooner), new features are offered in new product iterations. Many of these iterations are developed in parallel with each other. The work products, from the requirements to the code, must be easily maintained in order to keep to this rigorous schedule. Conforming to standards defined by the organization (and by international standards organizations) facilitates the making of products that are easily updated and maintained through each iteration. A general maintainability requirement must be defined at the owner level. As the relationships to other nonfunctional and functional focus areas are defined, the maintainability should be clarified as to its specific impact on the work products produced in those other focus areas. As the other requirements evolve, so should the maintainability requirement by describing the detail requirements as they relate to the structure and standard for the different perspectives and focus work products.

Project Constraints

Deadlines are not software requirements; they are project constraints. When attempting to be the first to market with a new concept, deadlines may be set in

Adding Community to the Infrastructure

105

stone rather than simply imposed arbitrarily. Another project-type constraint is the budget or venture capital dollars available to be used toward the development of the product. The amount of staff that can be hired to work on product development is also a constraint. These three examples are project constraints that have a direct impact on how the project is planned and tracked and on what can be developed by a specific point in time. When project-type constraints emerge, a negotiation process begins for what can feasibly be delivered for any given product.

Project constraints are important requirements to capture. As with functional and product constraints, project constraints also evolve through the different perspectives. Although the Internet program has one budget for resources, dollars, and time, each community is allocated its portion of the pot.

The project constraints are also important for other phases of development aside from requirements. Different choices can be made on hardware and network configurations based on the budget, staff, and time allocated to them. The project constraints impact those decisions.

Adding Community to the Infrastructure

The Zachman Information Systems Architecture (see Figure 3.7) shows how the framework works for software deliverables. Figure 3.8 shows the framework extended to support the important nonfunctional requirements. Notice that product and project constraints fit with the same perspectives as the six functional requirement types.

Although Figure 3.8 expands the number of focus areas compared with the Zachman framework, it still omits some of the different organizations involved in developing an Internet product. For example, to actually roll out an Internet application, other business areas are involved:

- *Advertising*: to develop the media representation for the product
- *Legal*: to develop product liability, copyright, and warranty documentation
- *Customer Service*: to develop processes to train persons assisting customers with questions, complaints, and returns

Each of these additional areas has its own process to provide the details for each perspective and focus. Each may have assigned a different project identifier, but the project objectives must satisfy the initiating requirements. Each has a different deliverable for each perspective and focus cell, organized as previously discussed for the information technology cells.









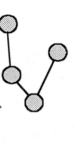





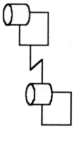





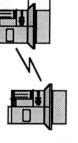





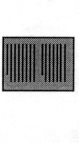





















	DATA	What	FUNCTION	How	NETWORK	Where	PEOPLE	Who	TIME	When	MOTIVATION	Why	
SCOPE (CONTEXTUAL)	List of Things Important to the Business		List of Processes the Business Performs		List of Locations in which the Business Operates		List of Organizations Important to the Business		List of Events Significant to the Business		List of Business Goals/Strat		SCOPE (CONTEXTUAL)
Planner	ENTITY = Class of Business Thing		Function = Class of Business Process		Node = Major Business Location		People = Major Organizations		Time = Major Business Event		Ends/Means=Major Bus. Goals Critical Success Factor		Planner
ENTERPRISE MODEL (CONCEPTUAL)	ENTITY = Semantic Model		e.g. Business Process Model		e.g. Business Logistics System		e.g. Work Flow Model		e.g. Master Schedule		e.g. Business Plan		ENTERPRISE MODEL (CONCEPTUAL)
Owner	Ent = Business Entity Rein = Business Relationship		Proc = Business Process I/O = Business Resources		Node = Business Location Link = Business Linkage		People = Organization Unit Work = Work Product		Time = Business Event Cycle = Business Cycle		End = Business Objective Means = Business Strategy		Owner
SYSTEM MODEL (LOGICAL)	e.g. Logical Data Model		e.g. Application Architecture		e.g. Distributed System Architecture		e.g. Human Interface Architecture		e.g. Processing Structure		e.g. Business Rule Model		SYSTEM MODEL (LOGICAL)
Designer	Ent = Data Entity Rein = Data Relationship		Proc = Application Function I/O = User Views		Node = I/O Function (Processor, Storage, etc.) Link = Line Characteristics		People = Role Work = Deliverable		Time = System Event Cycle = Processing Cycle		End = Structural Assertion Means = Action Assertion		Designer
TECHNOLOGY MODEL (PHYSICAL)	e.g. Physical Data Model		e.g. System Design		e.g. Technology Architecture		e.g. Presentation Architecture		e.g. Control Structure		e.g. Rule Design		TECHNOLOGY MODEL (PHYSICAL)
Builder	Ent = Segment/Table/etc. Rein = Pointer/Key/etc.		Proc = Computer Function I/O = Data Elements/Sets		Node = Hardware/System Link = Line Specifications		People = User Work = Screen Format		Time = Execute Cycle = Component Cycle		End = Condition Means = Action		Builder
DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)	e.g. Data Definition		e.g. Program		e.g. Network Architecture		e.g. Security Architecture		e.g. Timing Definition		e.g. Rule Specification		DETAILED REPRESENTATIONS (OUT-OF-CONTEXT)
Sub-Contractor	Ent = Field Rein = Address		Proc = Language Stmt I/O = Control Block		Node = Addresses Link = Protocols		People = Identity Work = Job		Time = Interrupt Cycle = Machine Cycle		End = Sub-condition Means = Step		Sub-Contractor
FUNCTIONING ENTERPRISE	e.g. DATA		e.g. FUNCTION		e.g. NETWORK		e.g. ORGANIZATION		e.g. SCHEDULE		e.g. STRATEGY		FUNCTIONING ENTERPRISE

Figure 3.7 John Zachman Information Systems Architecture for Software Perspectives and Focus (Source: John A. Zachman, Zachman International, 810-231-0531, www.zifa.com.)

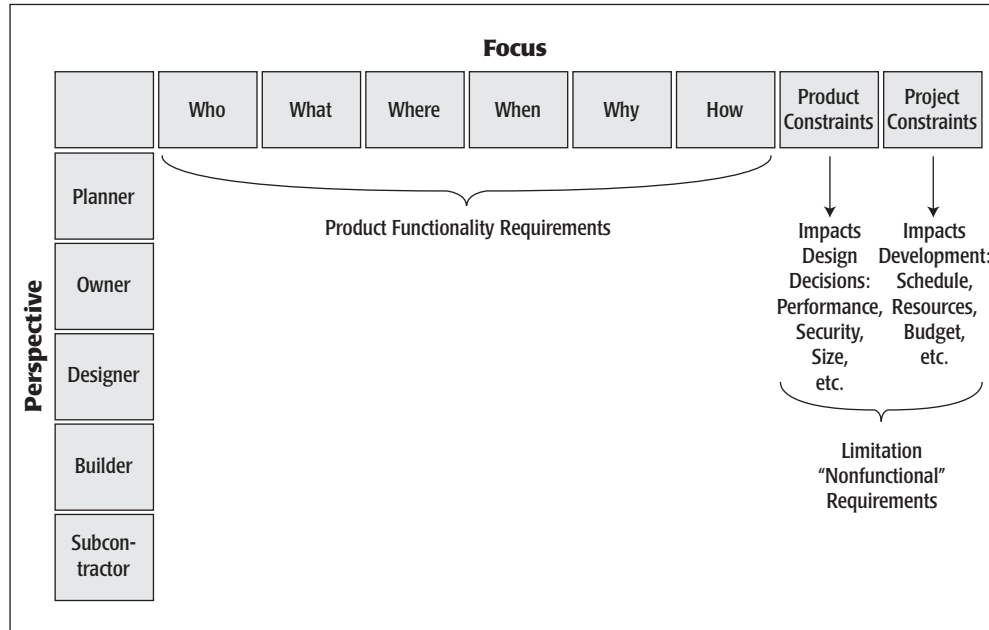


Figure 3.8 Zachman's Framework, Expanded Focus

With regards to the overall product plan, this creates a third dimension to the requirements set. This dimension, *community*, represents the different areas of business concentration. A community could be a business unit or a department of a business unit. The way in which a community is defined varies by organization and should be determined at the beginning of the program. The organization chart for your company is a good start for selecting communities. Here are some points to consider for communities:

- Product development and implementation require additional involvement by other organizations.
- Other organization areas will be dependent on perspective and focus requirements.
- Other organization areas will develop perspective and focus requirements that may impact software.
- Each organization area will have its own process that develops focus requirements through perspective levels.

Community changes the two-dimensional pattern shown in Figure 3.8 so that each requirement cell becomes multidimensional (see Figure 3.11 on page 111). As stated earlier in this chapter, we can categorize the communities into:

- Business practices (BP)
- Business support (BS)
- Business organization (BO)
- Information technology (IT)

Each of these areas can be broken into finer detail, depending on the deliverables that are required to develop a piece of the end product. For example, under business practice one may find marketing. Marketing may be further segmented into advertising and market research. Again, the breakdown will vary from organization to organization. No matter what level of breakdown, each subdivision has a separate deliverable and a different set of requirements it needs to satisfy to develop the complete end product. Each would follow the same objectives for each cell. To customize the framework for your business communities:

- Apply the organizational structure to the framework.
- Have each community represent specific areas of concentration that relate specifically to the organizational structure.
- Create Zachman's focus and perspectives for each organization or area initiating a subproject to satisfy the business objective.
- Add the two nonfunctional focus areas.
- Create embedded matrices of focus and perspective details specific to the community area.
- Create the association requirements to keep all cubes in sync in order to reach the common objectives of the product to be delivered.
- Derive all requirements from the initial business need (initiating requirement) or wants (the requirement request).

Requirement Associations

Requirements may not succeed all on their own. They need to be implemented with other requirements. For example, the process needs data to be able to perform its operations. For this reason, it is important to "link" or associate require-

ments that are dependent on each other. There is no need to define the data for both the “what” focus and the “how” focus. The data should be documented once and “used by” the process. To identify the process needs for the data, the data is associated to the process at the point it is needed. This builds a relationship requirement between data and process for specific purposes.

This association allows requirements to be normalized, in other words, to be stated once. One requirement may impact many requirements. For example, using the nonfunctional product constraints of reliability, a reliability requirement affects multiple “how” or “where” requirements. They need to be associated to build the quality requirement. Each association needs to be qualified in order to reflect the specific details of the reliability impact.

Just as any other requirement, association requirements must be satisfied to have a high-quality Internet product. These requirements are spawned by a specific requirement cell (perspective, focus, and community) to be satisfied by another area in order for them to work together. They are the glue that holds the different requirements together in a multidimensional model. (See Figure 3.9.)

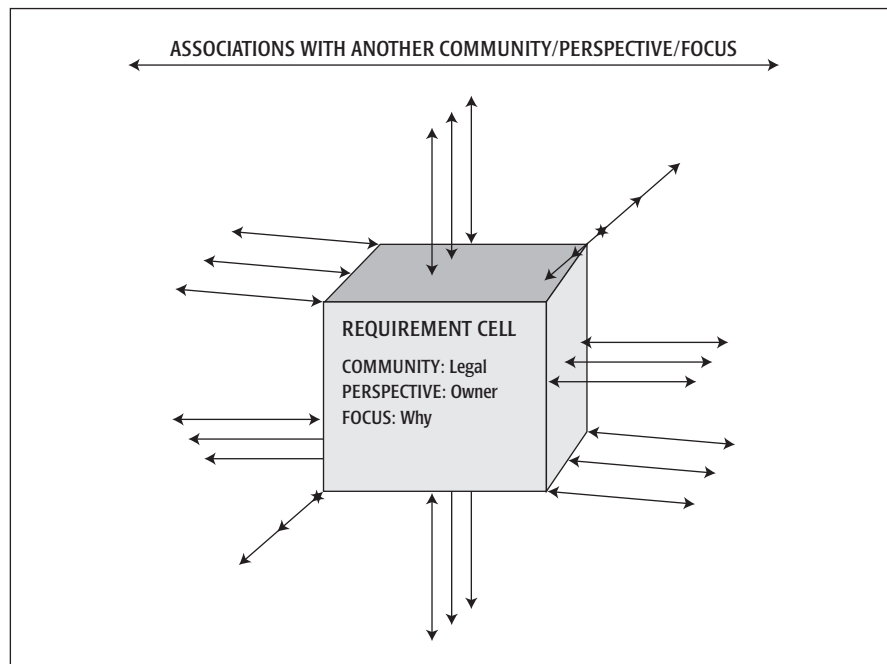


Figure 3.9 Potential Requirement Associations

Within each cell, specific needs must be addressed to refine the requirements set. For example, both the inventory subset of requirements and the billing subset of requirements can be defined separately. They may be two existing systems in larger brick-and-mortar type companies. In order to streamline the workflow between inventory management and billing, each system needs to be modified to work together. These special derived requirements are important for holding two separate requirements subsets (billing and inventory) together.

When one requirement needs to use (or reference) information captured in another cell, it is important to build a link between them. Most CASE tools do this automatically. This creates a “usage trail” so that if either requirement changes during the development process, you can easily identify the potential impact on other requirements. All affected requirements (the two independent and the one relationship) need to be implemented in order to have a successful Internet system. They can be tested and implemented independently. (See Figure 3.10.)

In Figure 3.11 the Zachman framework is extended, showing how the relationships (associations) form the mortar between the requirement cells.

Two pieces of information are required in order for the association requirement to be connected. One is the actual content; the other is the usage trail (link). The usage trail is the identifier of both requirements that are being associated.

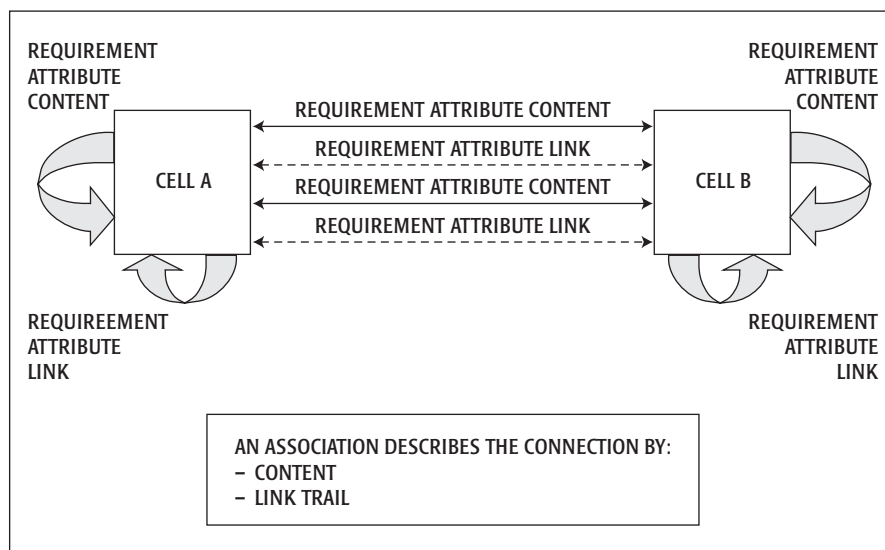


Figure 3.10 Internal and External Cell Association

Requirement Associations

111

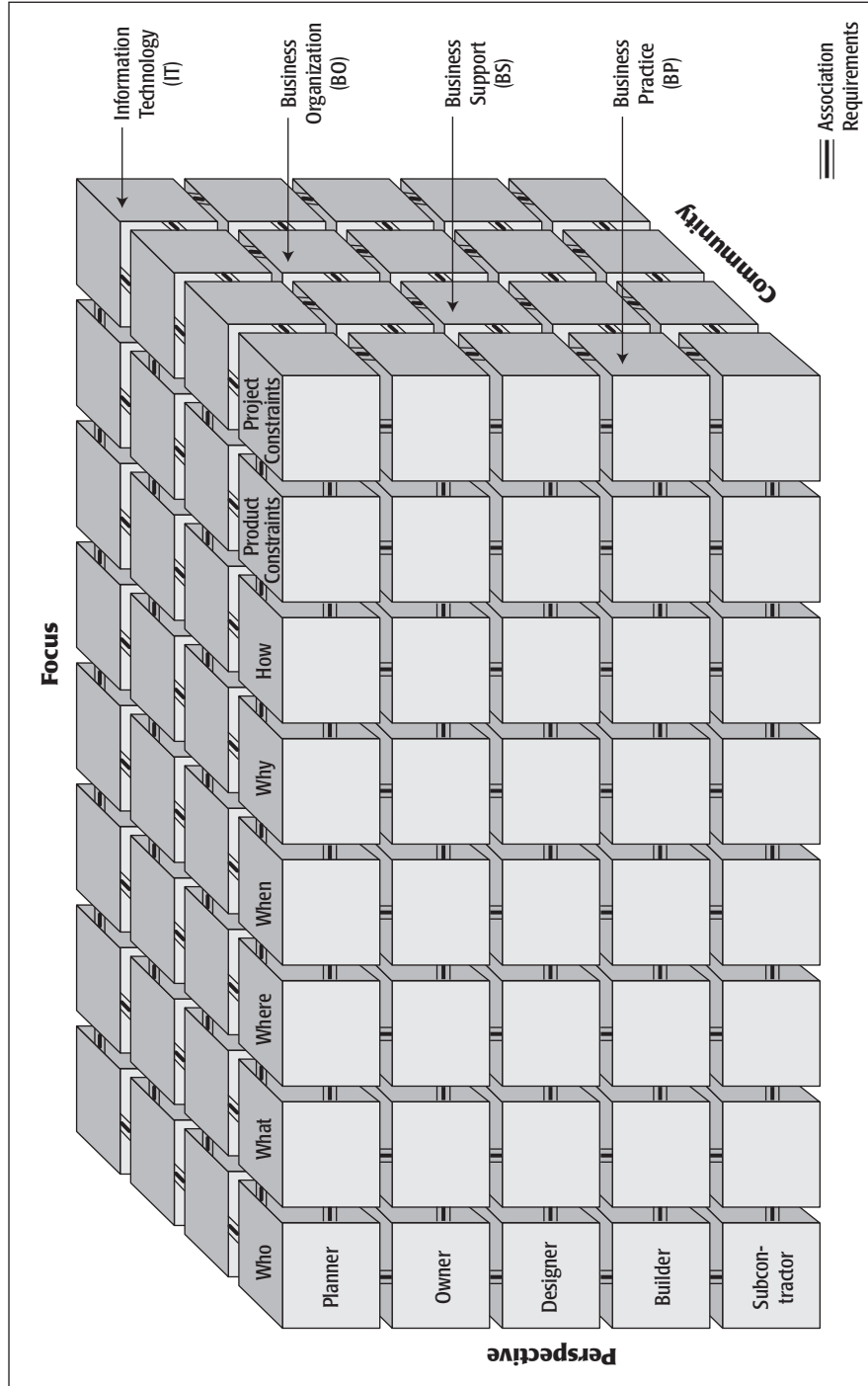


Figure 3.11 Requirements Set Framework Showing Perspective and Focus Categories

These types of requirements are often overlooked. Once a requirement is allocated to a specific community, the community may satisfy its requirements without thinking of the possible impact the requirement may have on another community. The need for these types of requirements is finally realized somewhere down the product's development path. The result is that the other areas are playing catch-up or that both areas are modifying the design to accommodate the interaction. The modification is usually not ideal and could have been accounted for earlier in the development cycle. The catch-up is costly in terms of time and money.

To account for these association requirements, the set of requirements should be available for all communities, focus areas, and perspectives. As the requirements are being written for the specific cell, the requirements engineer should ask what needs to be done for other cells to succeed in satisfying this requirement. As existing requirements for other cells are reviewed, it is important to ask those same questions. If one cell needs to be associated with another cell, what additional requirements need to be developed to facilitate the association?

Some of the support associations will be noted by the individual community as its "who" or association requirement. Others will not. For example, network planning is often forgotten until a month before the software product is to be implemented. The network-planning people need information that should have been gathered early in the program. Unfortunately, the program ends up delayed or the network personnel run ragged just to keep to the deadline.

Organization Impact

If nothing else, the expanded Zachman framework helps us understand that there are many requirements to be captured in order to build any product, including an Internet product. With this number of requirements, it is important to organize them in a fashion that makes it easy to identify any gaps in knowledge. Adding community, perspective, focus, and association as another means to identify categories of requirements assists in organizing the requirements for gap and inconsistency analysis.

Each perspective for each community creates a separate work product. A separate work product may be (but is not necessarily) further delivered for each focus and for each perspective and each community. Each of these work product deliverables contains requirements that should be linked to related requirements in other documents. This is easier to accomplish with the assistance of a requirements management tool.

Conclusion

113

Most requirements management tools can support the framework. Implementing the requirements set framework requires a requirements engineer who possesses a full understanding of the requirements set framework and the requirements process (including allocation) to define the filter process for each requirement. The benefits of implementing the framework in the requirements management tool are (1) easier validation and (2) a reporting process between perspectives and iterations of the Internet product.

CASE tools are available that capture some of the categories of requirements. None of them captures all the different types of requirements. Each of these tools can provide a documented output of the requirements for its concentrated focus and perspective. This document should be fed into the requirements management tool for gap and consistency analysis as well as for change control within the entire requirements set.

Conclusion

This chapter introduced how the different types of requirements can be organized. Categorizing the requirements makes it easier to develop the question list as well as to identify gaps in knowledge. The objective is to ensure that the product to be developed is fully understood from all angles. Adding the question list and process to the organizational structure turns the framework into a pattern (see Appendix B).

Perspectives reflect the areas of responsibility for each role. For example, a designer's perspective is quite different from the owner's perspective; the builder's perspective is different from a designer's. Each evolves the requirements by providing details needed to interpret his perspective. The owner's perspective must be satisfied in the final product. Though each role has a different perspective, all roles work together as a team to build the quality product.

When building a home, each perspective requires a different focus or view. These views concentrate on a specific area for that perspective. For example, the house requires a plumber's view, an electrician's view, a landscaper's view, and so forth. Each has similar deliverables: scope, owner's model, physical model, specifications, components, and the final product. The deliverables at each perspective level have their own focus areas.

In software engineering terms, the focus includes both functional and non-functional requirements. Functional requirements are the "who" (interface associations), "what" (data), "where" (networks), "when" (events), "why" (business

rules), and “how” (processes). The nonfunctional requirements are the product and project constraints. Similarly, each perspective has eight different views, one for each of these eight areas.

It is important to realize that nothing can be viewed independently. All focuses and perspectives are important for building a high-quality product. Each view must drill down a concept into finite detail to ensure compliance and quality and to meet the business needs. Each focus and perspective requires different probing questions.

Linking the different focuses and perspectives requires additional association or support-type requirements that will tie all intersecting squares. For example, when building a home, electricians need to ensure they do not affect the plumber’s requirements. This creates additional association requirements for the electrician and plumber. In system engineering terms, association requirements are created for hardware and software to work together. Probing questions are required to tie the focus and perspective to other focus areas and perspectives.

We need to expand our view of requirements beyond software-implemented requirements. Additional dimensions need to be included to incorporate all business areas that may be affected by the business request. This includes such business areas as legal and human resources. Each business area will react in some way to the original requirement. Each business area will have its own process for satisfying a specific need. Each business community may spawn additional IT requirements. These may initiate additional software projects that enhance existing systems (for example, change billing system) or require maintaining data contents (for example, add a new product number), or they may initiate non-IT projects (for example, modify legal contracts).

The Zachman framework can be expanded to include the additional dimension of community to support each business area. Each community has its own focus and perspective. Each multidimensional cube requires additional association requirements to support the linkage of the requirements.

The requirements that fall into each community, perspective, focus, and association create a full requirements set. A requirements set contains all the individual requirements and related information associated with a business objective. Some general guidelines follow:

- Each requirement should be stated only once.
- Requirements should not overlap, that is, they should not refer to other requirements or the capabilities of other requirements.

Conclusion

115

- Explicit relationships should be defined among individual requirements to show how the requirements are related to or dependent on each other to form a complete product.
- The requirements set should include all requirements needed by the business user as well as those needed for the definition of the product.
- The requirements set should be consistent and noncontradictory in the level of detail between formats.
- Community and perspective should be used to subdivide the requirements set. Though many different groups may develop specific focuses, this is not recommended since it increases the risk of missing association requirements or requirements that tie focus requirements with other focus areas.
- A skilled requirements engineer should be involved in implementing the framework into a requirements management tool. This includes the transition from different CASE tools and other formats into the requirements management tool.

Adding a process for filling each cell, as well as filter rules for each cell, turns the framework into a pattern. This requirements pattern is a meta pattern that can be tailored to any type of application. The next chapter does just this: it takes the requirements set framework and discusses the objectives and the types of questions that should be asked to capture the needs of the Internet product.

