IMS564 | USER EXPERIENCE DESIGN

THE SCOPE PLANE

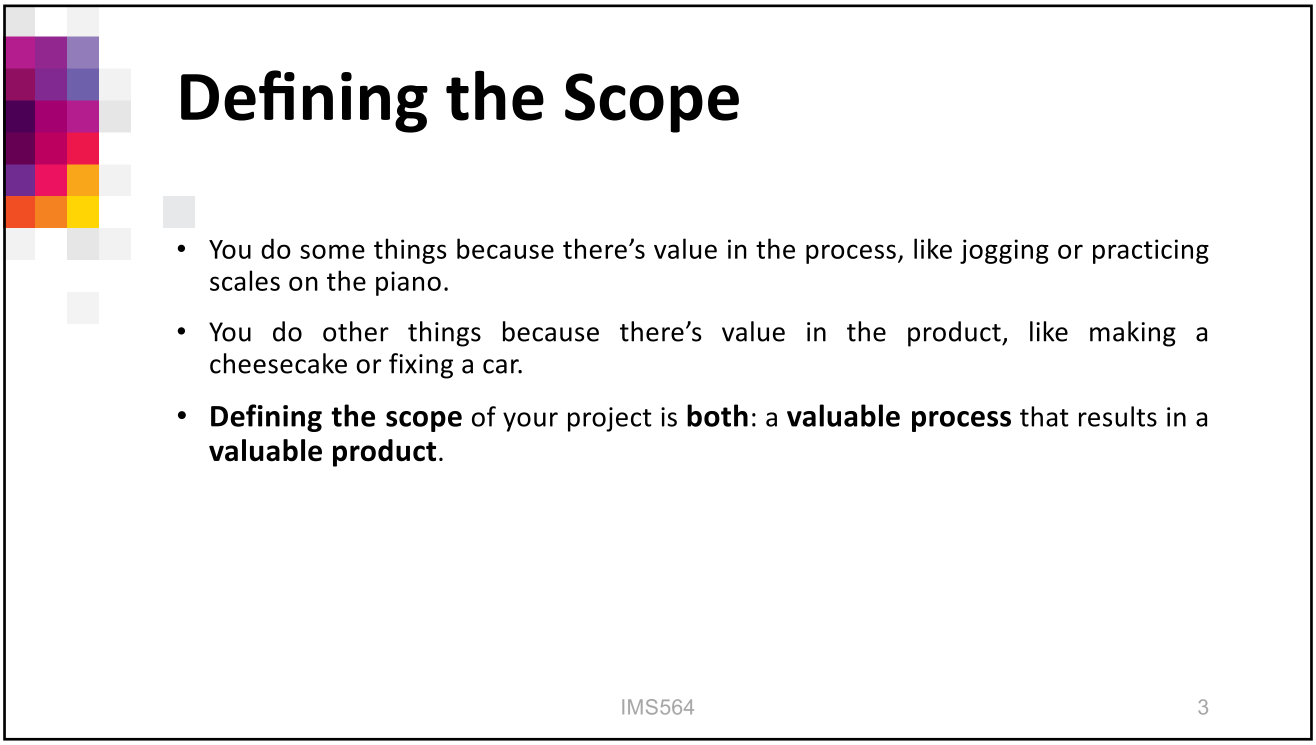
Dr Norizan Anwar

1

Table of Content

- Defining the Scope
- Functionality and Content
- Defining Requirements
- Functionality Specifications
- Content Requirements
- Prioritizing Requirements

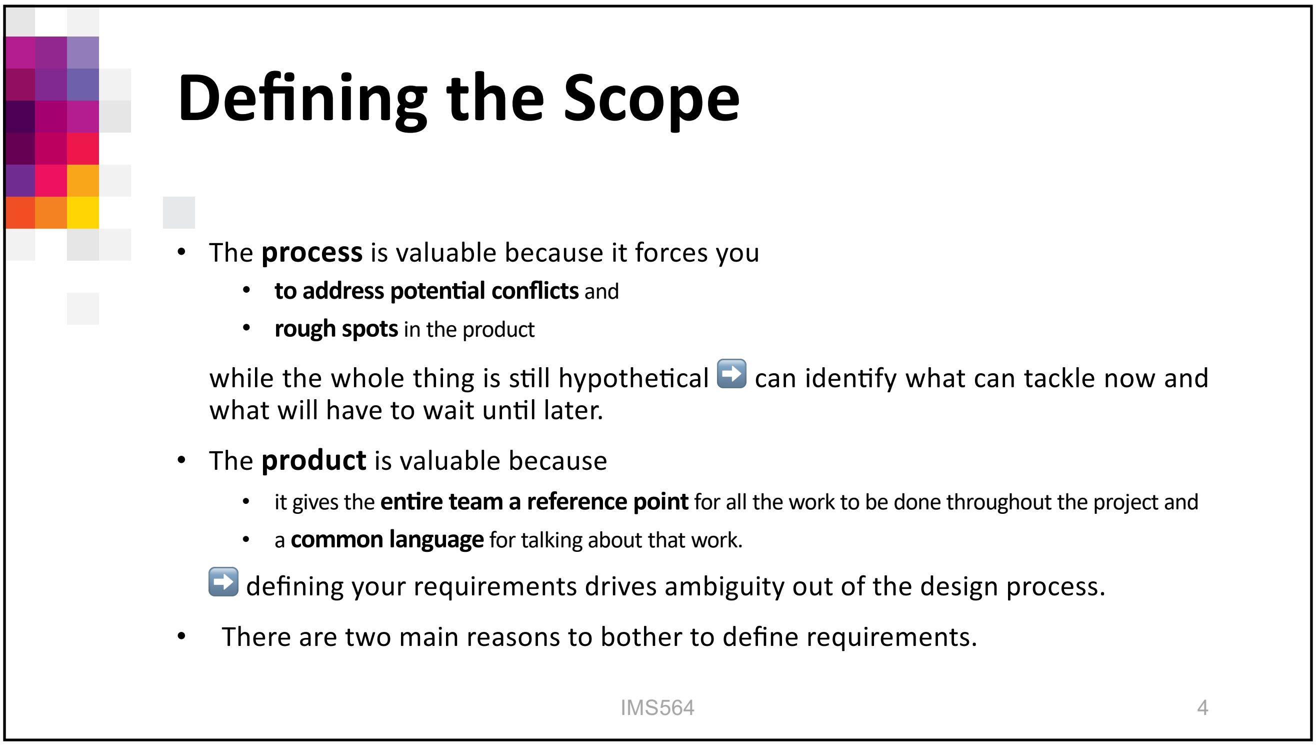
2



Defining the Scope

- You do some things because there’s value in the process, like jogging or practicing scales on the piano.
- You do other things because there’s value in the product, like making a cheesecake or fixing a car.
- **Defining the scope** of your project is **both**: a **valuable process** that results in a **valuable product**.

3



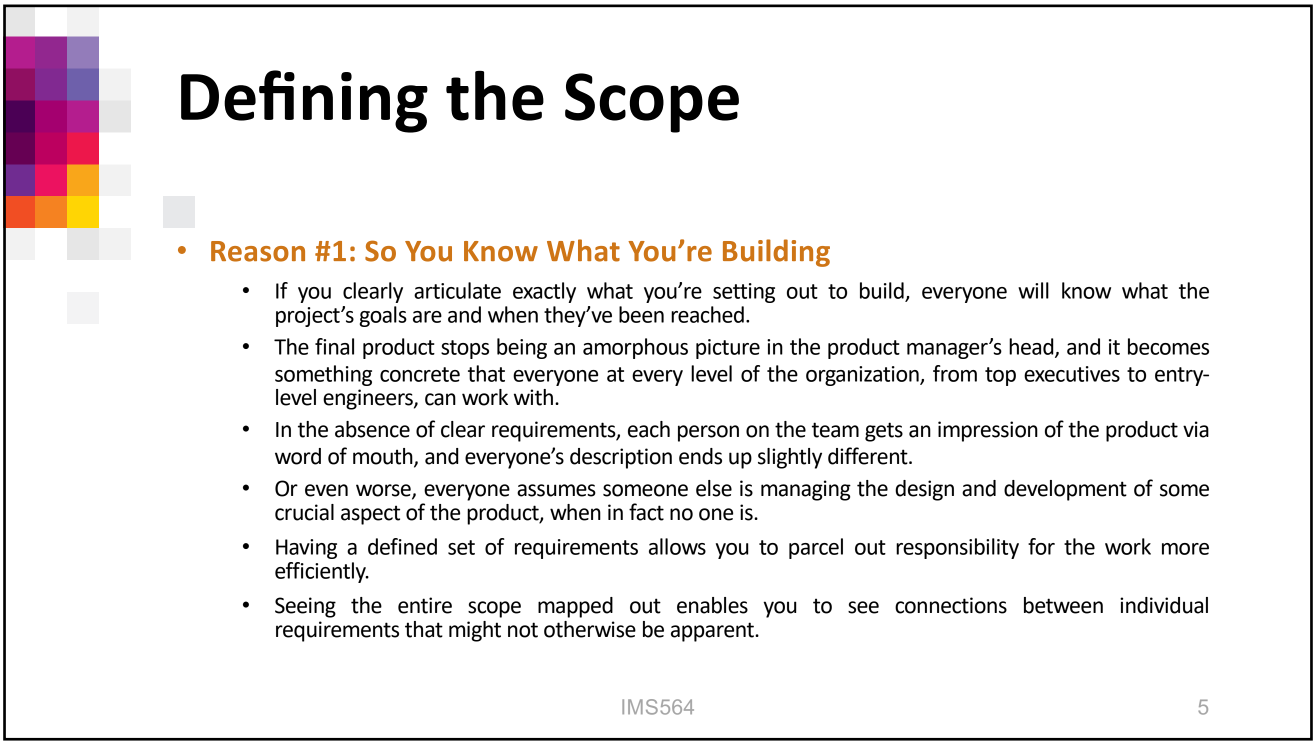
Defining the Scope

- The **process** is valuable because it forces you
  - **to address potential conflicts** and
  - **rough spots** in the product

  while the whole thing is still hypothetical ➡ can identify what can tackle now and what will have to wait until later.
- The **product** is valuable because
  - it gives the **entire team a reference point** for all the work to be done throughout the project and
  - a **common language** for talking about that work.

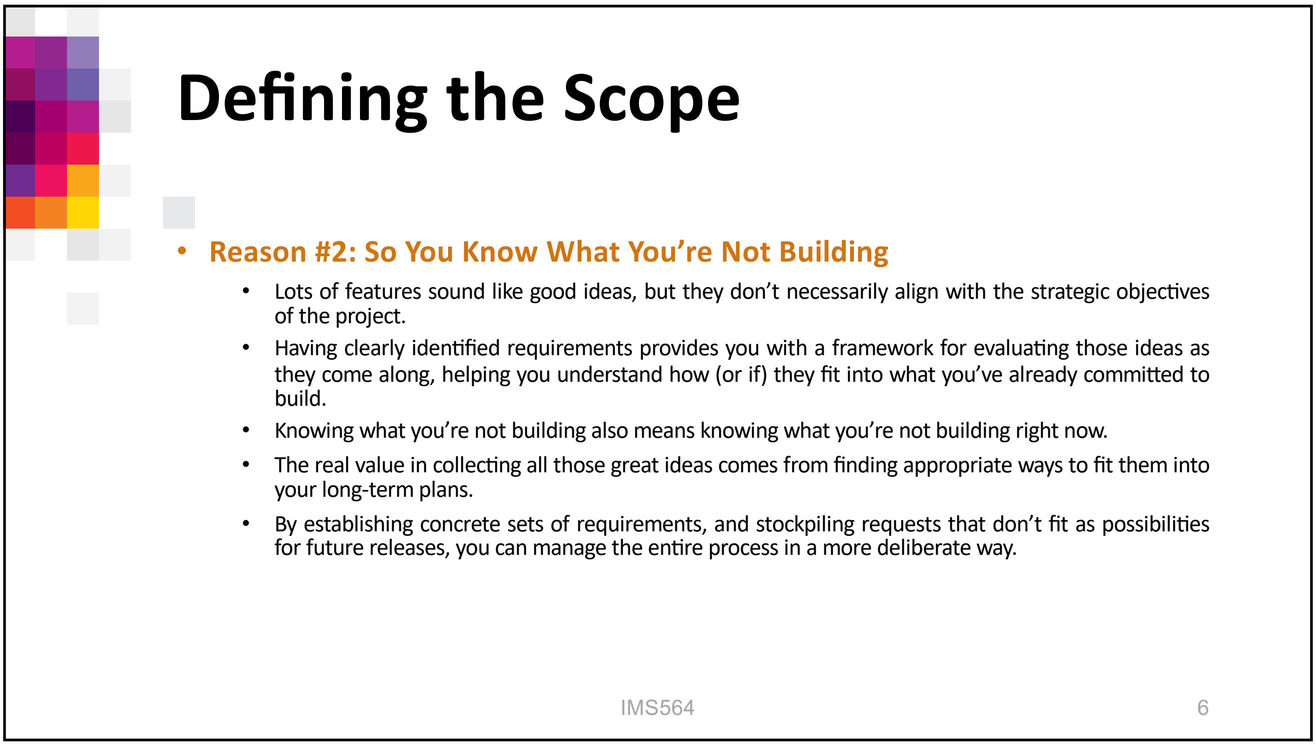
  ➡ defining your requirements drives ambiguity out of the design process.
- There are two main reasons to bother to define requirements.

4



Defining the Scope

- **Reason #1: So You Know What You're Building**
  - If you clearly articulate exactly what you're setting out to build, everyone will know what the project's goals are and when they've been reached.
  - The final product stops being an amorphous picture in the product manager's head, and it becomes something concrete that everyone at every level of the organization, from top executives to entry-level engineers, can work with.
  - In the absence of clear requirements, each person on the team gets an impression of the product via word of mouth, and everyone's description ends up slightly different.
  - Or even worse, everyone assumes someone else is managing the design and development of some crucial aspect of the product, when in fact no one is.
  - Having a defined set of requirements allows you to parcel out responsibility for the work more efficiently.
  - Seeing the entire scope mapped out enables you to see connections between individual requirements that might not otherwise be apparent.

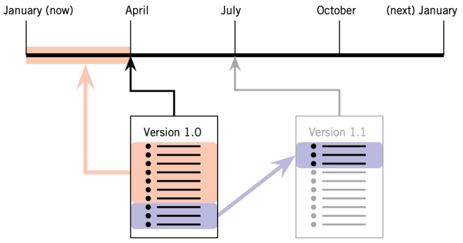


Defining the Scope

- **Reason #2: So You Know What You're Not Building**
  - Lots of features sound like good ideas, but they don't necessarily align with the strategic objectives of the project.
  - Having clearly identified requirements provides you with a framework for evaluating those ideas as they come along, helping you understand how (or if) they fit into what you've already committed to build.
  - Knowing what you're not building also means knowing what you're not building right now.
  - The real value in collecting all those great ideas comes from finding appropriate ways to fit them into your long-term plans.
  - By establishing concrete sets of requirements, and stockpiling requests that don't fit as possibilities for future releases, you can manage the entire process in a more deliberate way.

Defining the Scope

- **Reason #2: So You Know What You're Not Building**



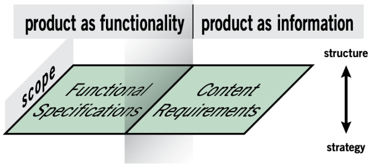
Requirements that can't be met in the current schedule can form the basis for the next milestone in your development cycle.

- If you don't consciously manage your requirements, you'll get caught in the dreaded "scope creep."
- Each additional requirement may not seem like that much extra work.
- But put them all together, and you've got a project rolling away out of control, crushing deadlines and budget estimates on its way toward an inevitable final crash.

Functionality and Content

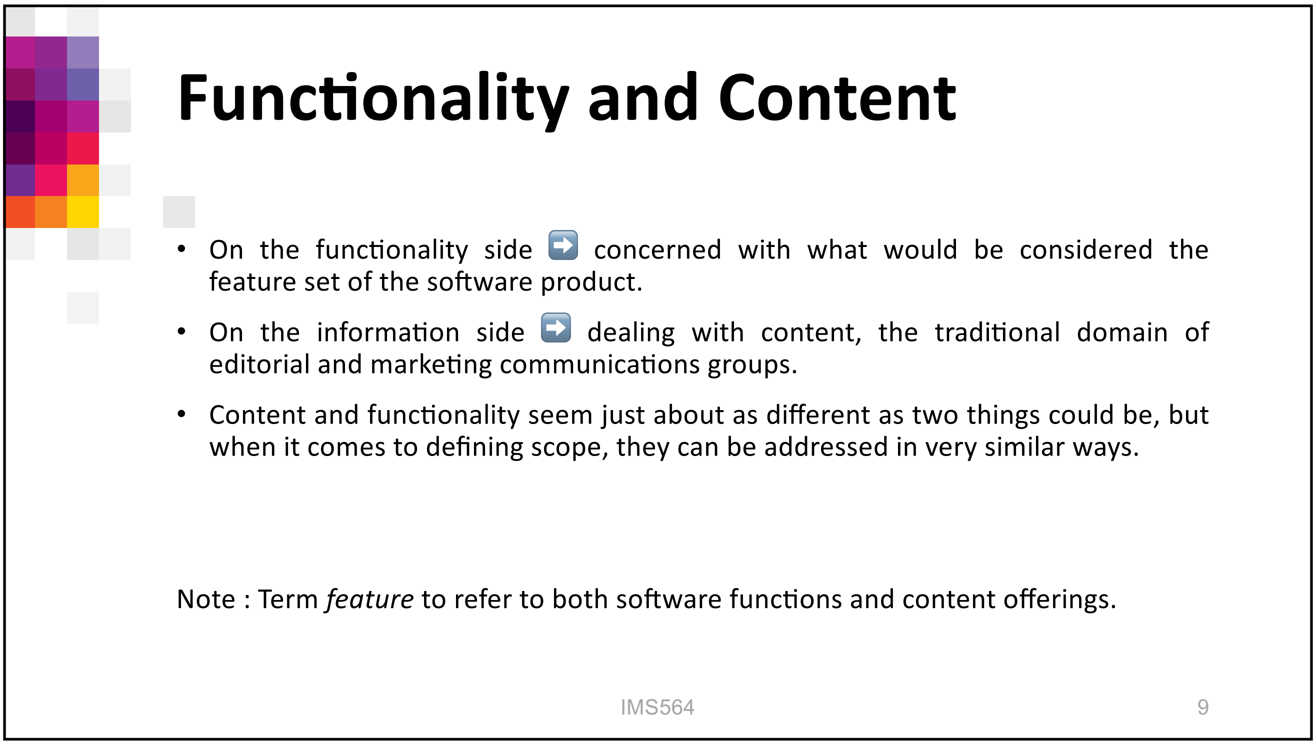
- Start from the abstract question of
  - "Why are we making this product?" | The strategy plane

    ⬇
  - "What are we going to make?" | The scope plane



- On the scope plane, the split between
  - the Web as a vehicle for functionality and
  - the Web as an information medium

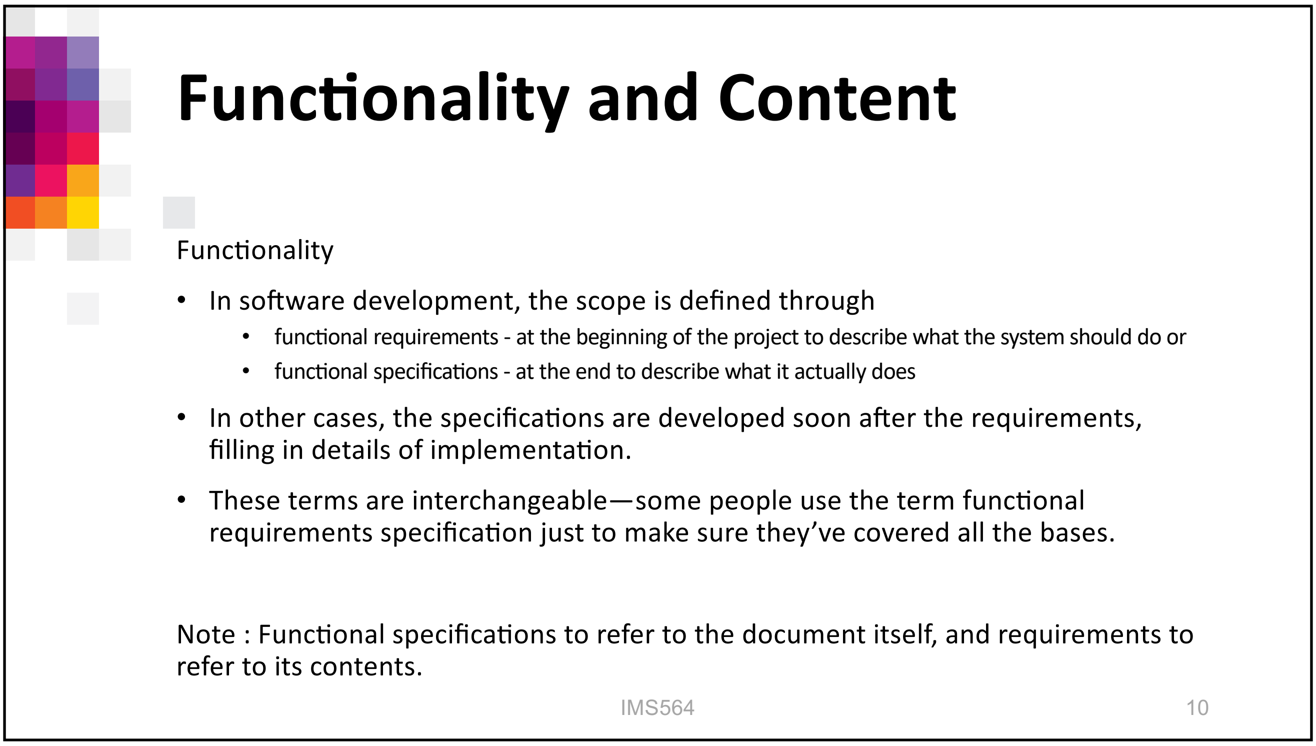
starts coming into play.



Functionality and Content

- On the functionality side ➡ concerned with what would be considered the feature set of the software product.
- On the information side ➡ dealing with content, the traditional domain of editorial and marketing communications groups.
- Content and functionality seem just about as different as two things could be, but when it comes to defining scope, they can be addressed in very similar ways.

Note : Term *feature* to refer to both software functions and content offerings.



Functionality and Content

Functionality

- In software development, the scope is defined through
  - functional requirements - at the beginning of the project to describe what the system should do or
  - functional specifications - at the end to describe what it actually does
- In other cases, the specifications are developed soon after the requirements, filling in details of implementation.
- These terms are interchangeable—some people use the term functional requirements specification just to make sure they've covered all the bases.

Note : Functional specifications to refer to the document itself, and requirements to refer to its contents.

Functionality and Content

Content

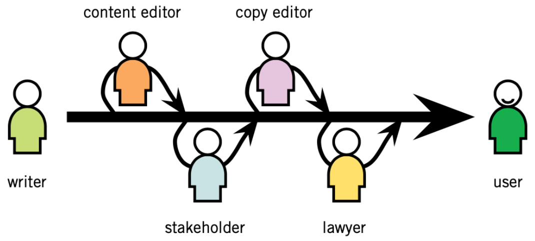
- Content development often involves a less formal requirements - definition process than software does, but the underlying principles are the same.
- A content developer will sit down and talk with people or pore over source material, whether that be a database or a drawer full of news clippings, in order to determine what information needs to be included in the content he's/she's developing.
- This process for defining content requirements is actually not all that different from the technologist brainstorming features with stakeholders and reviewing existing documentation. The purposes and approaches are the same.
- Content requirements often have functional implications.

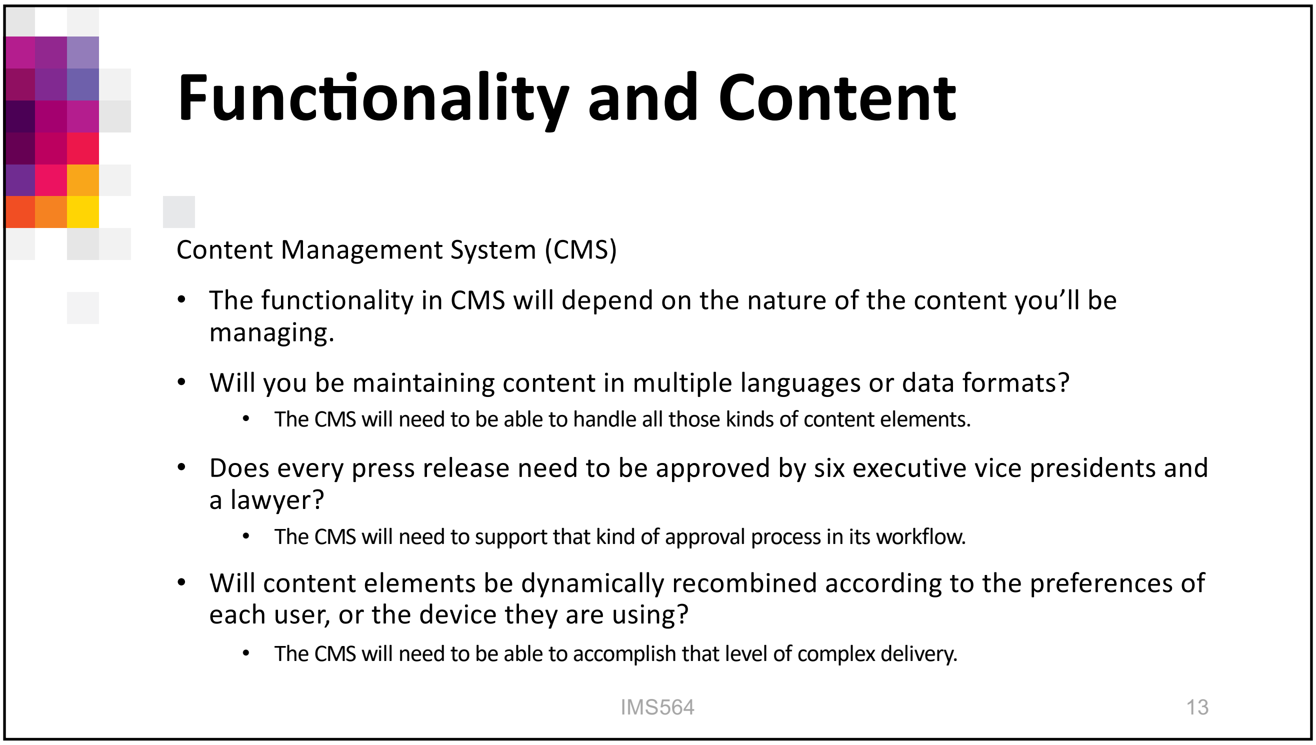
Functionality and Content

Content Management System (CMS)

- CMS come in all shapes and sizes, from very large and complex systems that dynamically generate pages from a dozen different data sources to lightweight tools optimized for managing one specific type of content feature in the most efficient way.

A content management system can automate the workflow required to produce and deliver content to users.

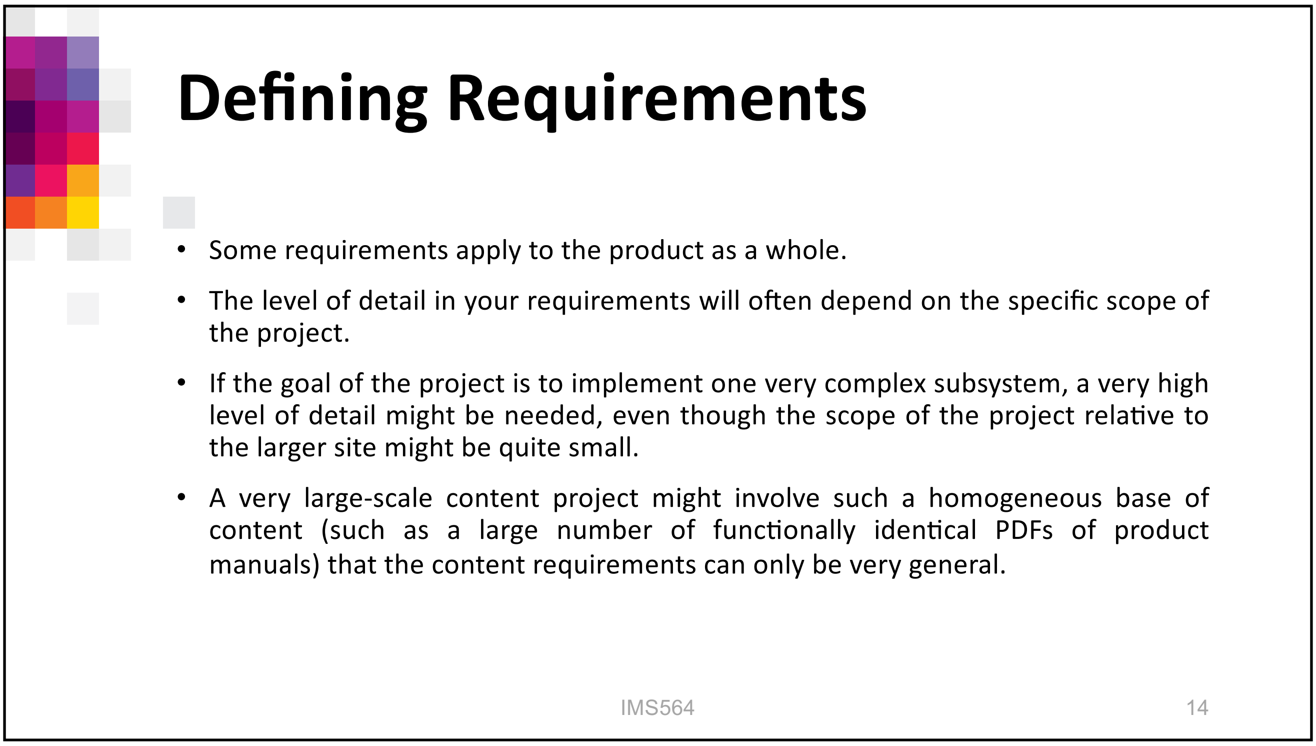




Functionality and Content

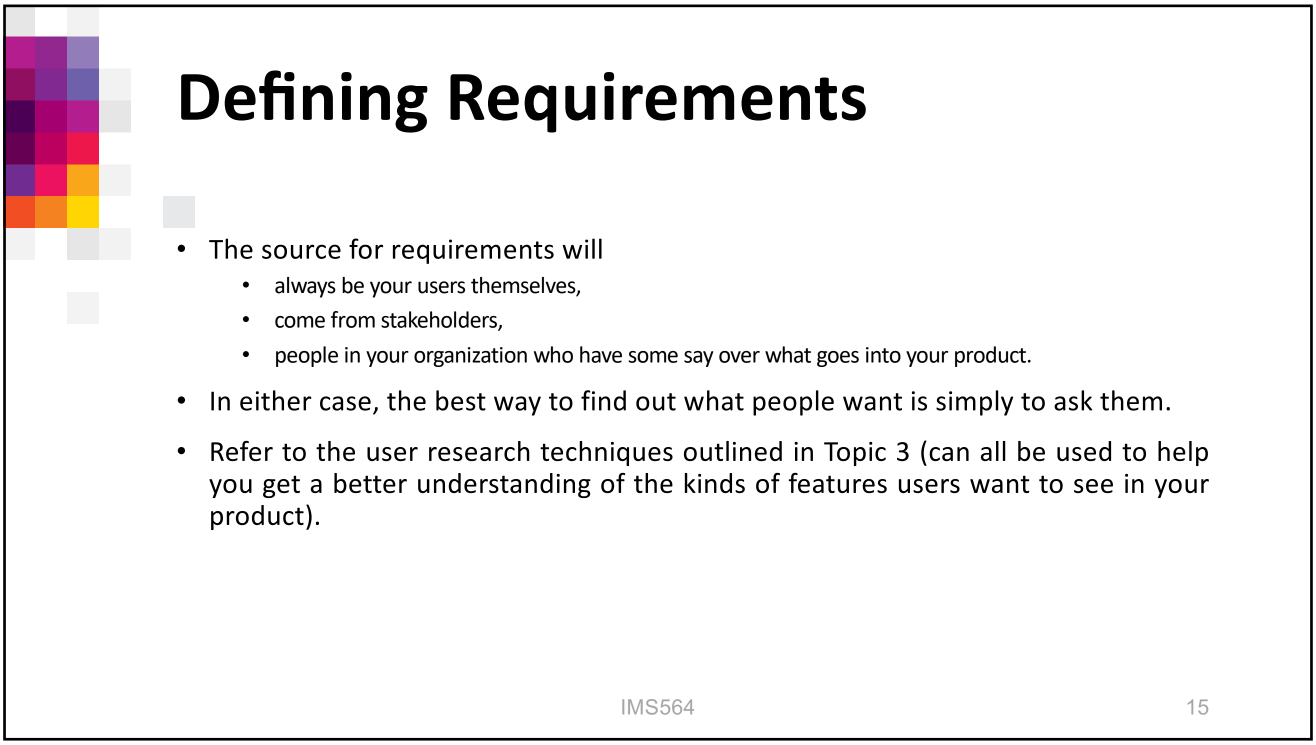
Content Management System (CMS)

- The functionality in CMS will depend on the nature of the content you'll be managing.
- Will you be maintaining content in multiple languages or data formats?
  - The CMS will need to be able to handle all those kinds of content elements.
- Does every press release need to be approved by six executive vice presidents and a lawyer?
  - The CMS will need to support that kind of approval process in its workflow.
- Will content elements be dynamically recombined according to the preferences of each user, or the device they are using?
  - The CMS will need to be able to accomplish that level of complex delivery.



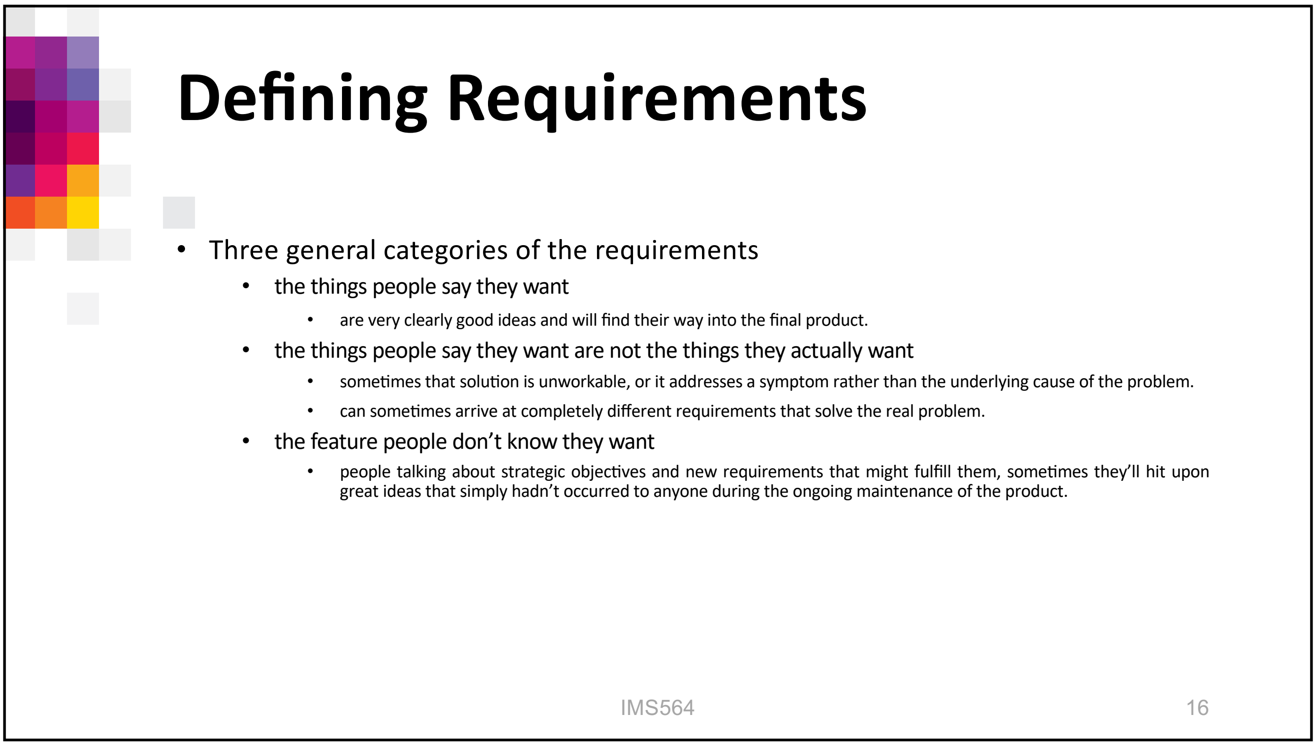
Defining Requirements

- Some requirements apply to the product as a whole.
- The level of detail in your requirements will often depend on the specific scope of the project.
- If the goal of the project is to implement one very complex subsystem, a very high level of detail might be needed, even though the scope of the project relative to the larger site might be quite small.
- A very large-scale content project might involve such a homogeneous base of content (such as a large number of functionally identical PDFs of product manuals) that the content requirements can only be very general.



Defining Requirements

- The source for requirements will
  - always be your users themselves,
  - come from stakeholders,
  - people in your organization who have some say over what goes into your product.
- In either case, the best way to find out what people want is simply to ask them.
- Refer to the user research techniques outlined in Topic 3 (can all be used to help you get a better understanding of the kinds of features users want to see in your product).



Defining Requirements

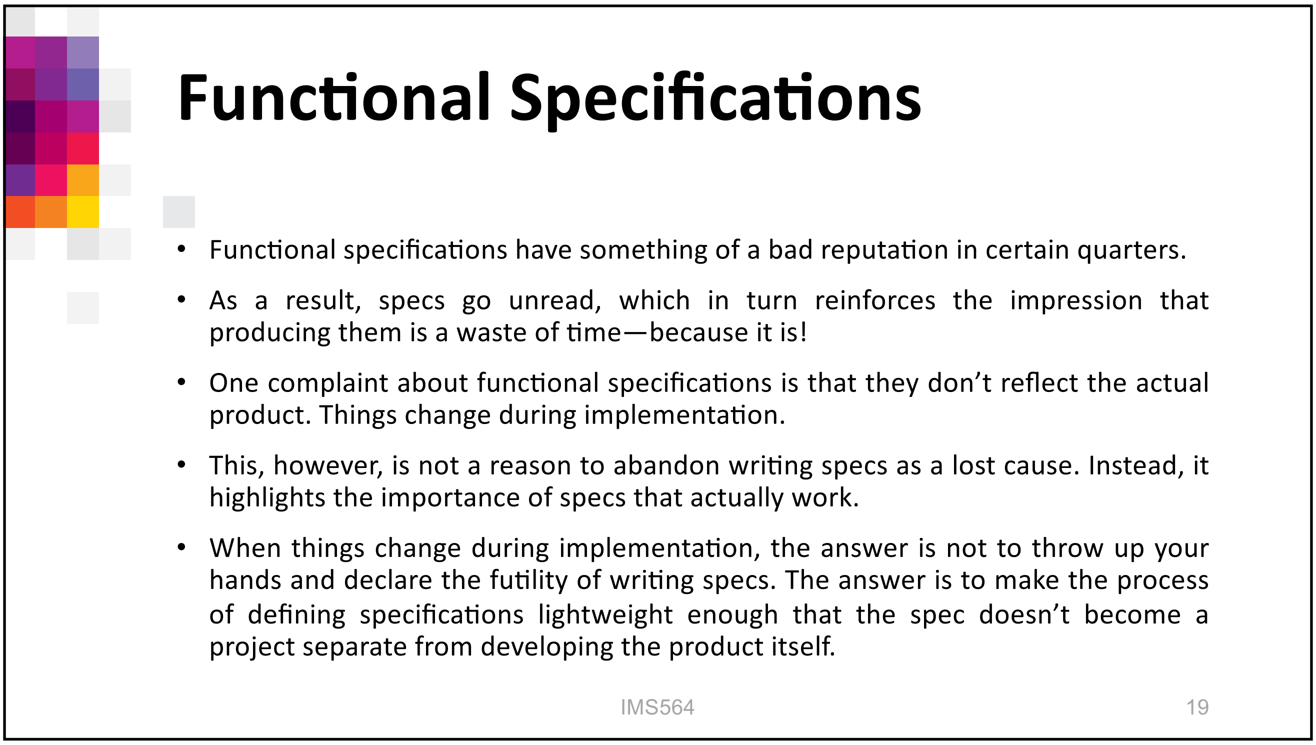
- Three general categories of the requirements
  - the things people say they want
    - are very clearly good ideas and will find their way into the final product.
  - the things people say they want are not the things they actually want
    - sometimes that solution is unworkable, or it addresses a symptom rather than the underlying cause of the problem.
    - can sometimes arrive at completely different requirements that solve the real problem.
  - the feature people don't know they want
    - people talking about strategic objectives and new requirements that might fulfill them, sometimes they'll hit upon great ideas that simply hadn't occurred to anyone during the ongoing maintenance of the product.

Defining Requirements

- People most deeply involved in creating and working with a product are the ones least able to imagine new directions for it.
- When you spend all your time immersed in maintaining an existing product, you can often forget which of your constraints are real, and which are simply products of historical choices.
- For this reason, group brainstorming sessions that bring together people from diverse parts of the organization or represent diverse user groups can be very effective tools in opening the minds of participants to possibilities they wouldn't have considered before.
- Getting an engineer, a customer service agent, and a marketing person in a room together to talk about the same Web site can be enlightening for everyone.

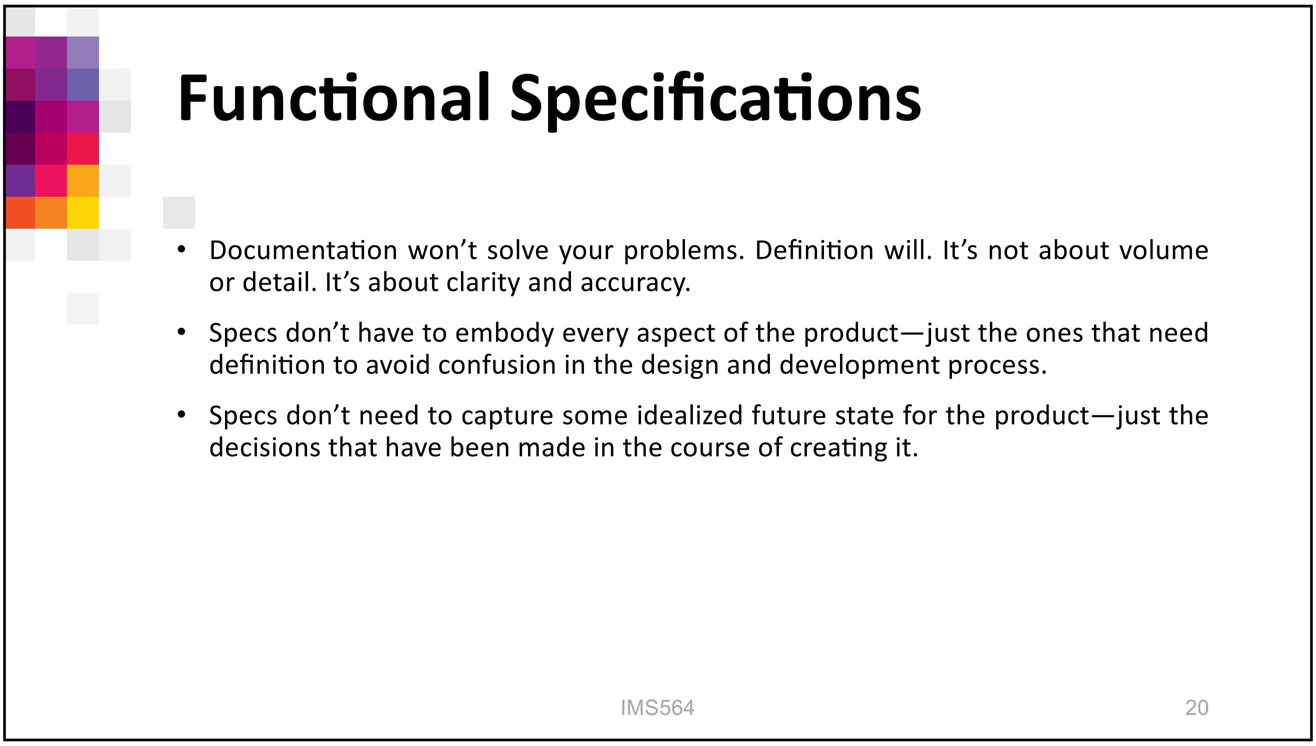
Defining Requirements

- Look at the technique of creating fictional characters called personas to help you better understand user needs. (Refer to Topic 4)
  - Use those personas again by putting your fictional characters into little stories called scenarios.
  - A scenario is a short, simple narrative describing how a persona might go about trying to fulfill one of those user needs.
  - By imagining the process of your users might go through, you can come up with potential requirements to help meet their needs.
- Look to your competitors for inspiration.
  - Anyone else in the same business is almost certainly trying to meet the same user needs and is probably trying to accomplish similar product objectives as well.
  - Has a competitor found a particularly effective feature to meet one of these strategic objectives?
  - How have they addressed the same trade-offs and compromises we face?



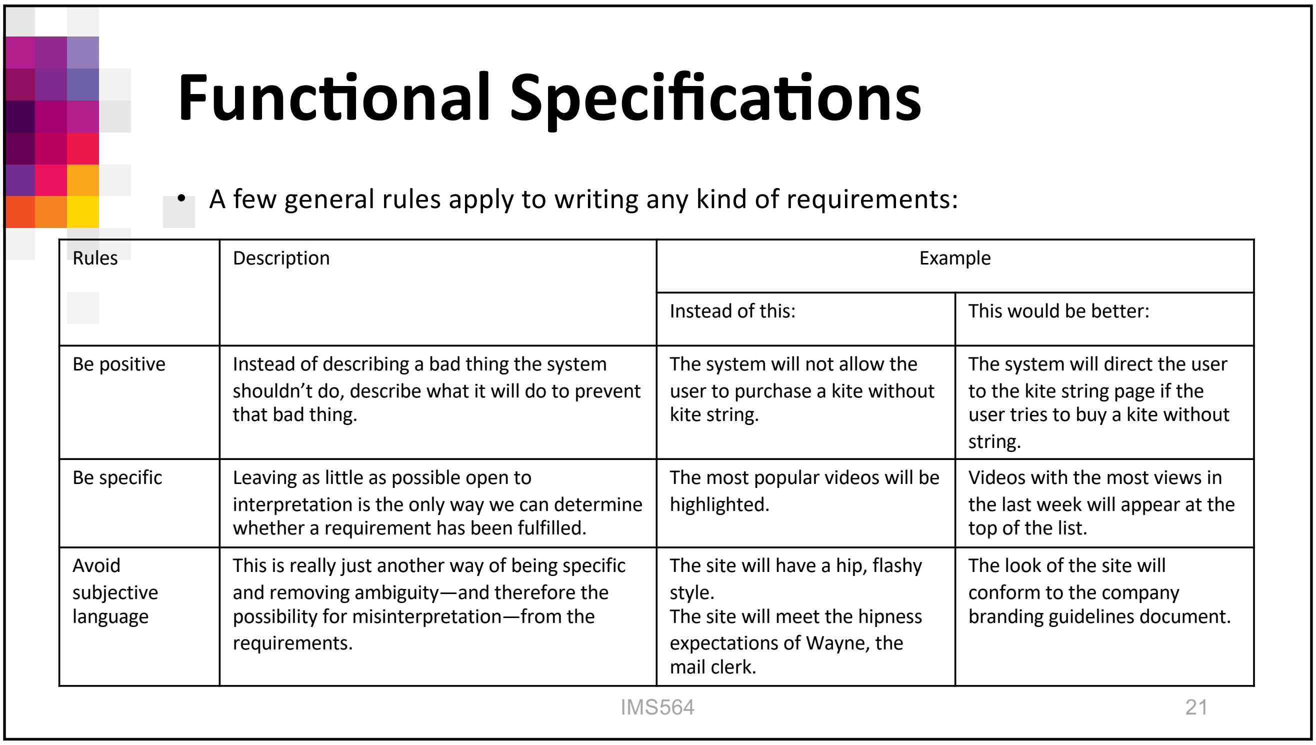
Functional Specifications

- Functional specifications have something of a bad reputation in certain quarters.
- As a result, specs go unread, which in turn reinforces the impression that producing them is a waste of time—because it is!
- One complaint about functional specifications is that they don't reflect the actual product. Things change during implementation.
- This, however, is not a reason to abandon writing specs as a lost cause. Instead, it highlights the importance of specs that actually work.
- When things change during implementation, the answer is not to throw up your hands and declare the futility of writing specs. The answer is to make the process of defining specifications lightweight enough that the spec doesn't become a project separate from developing the product itself.



Functional Specifications

- Documentation won't solve your problems. Definition will. It's not about volume or detail. It's about clarity and accuracy.
- Specs don't have to embody every aspect of the product—just the ones that need definition to avoid confusion in the design and development process.
- Specs don't need to capture some idealized future state for the product—just the decisions that have been made in the course of creating it.

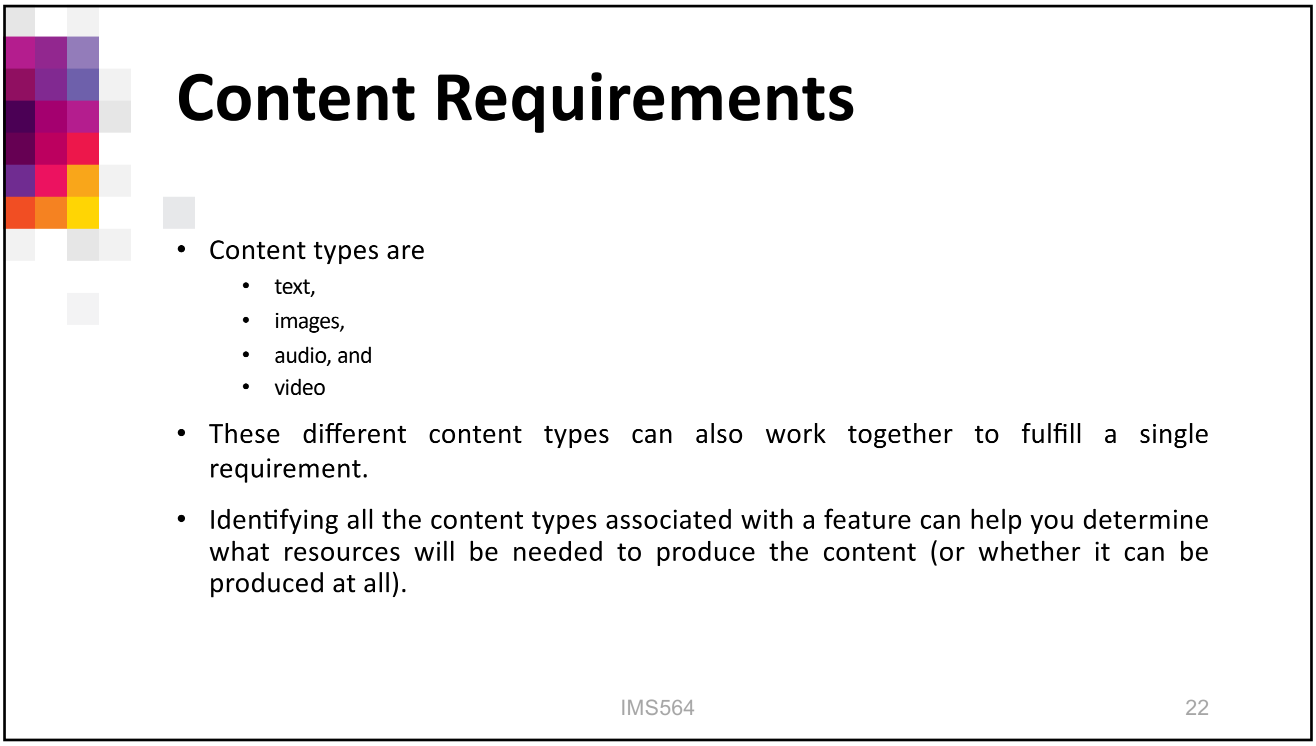


Functional Specifications

- A few general rules apply to writing any kind of requirements:

Rules	Description	Example	
		Instead of this:	This would be better:
Be positive	Instead of describing a bad thing the system shouldn't do, describe what it will do to prevent that bad thing.	The system will not allow the user to purchase a kite without kite string.	The system will direct the user to the kite string page if the user tries to buy a kite without string.
Be specific	Leaving as little as possible open to interpretation is the only way we can determine whether a requirement has been fulfilled.	The most popular videos will be highlighted.	Videos with the most views in the last week will appear at the top of the list.
Avoid subjective language	This is really just another way of being specific and removing ambiguity—and therefore the possibility for misinterpretation—from the requirements.	The site will have a hip, flashy style. The site will meet the hipness expectations of Wayne, the mail clerk.	The look of the site will conform to the company branding guidelines document.

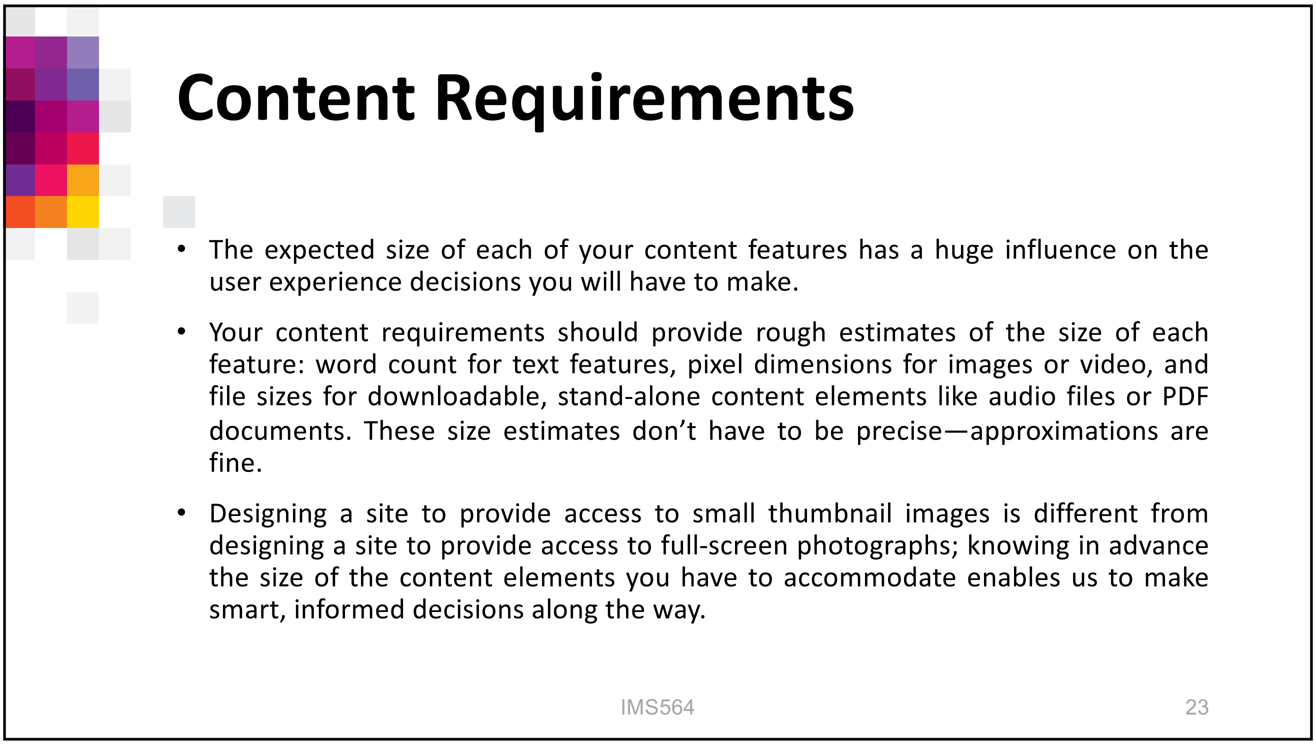
21



Content Requirements

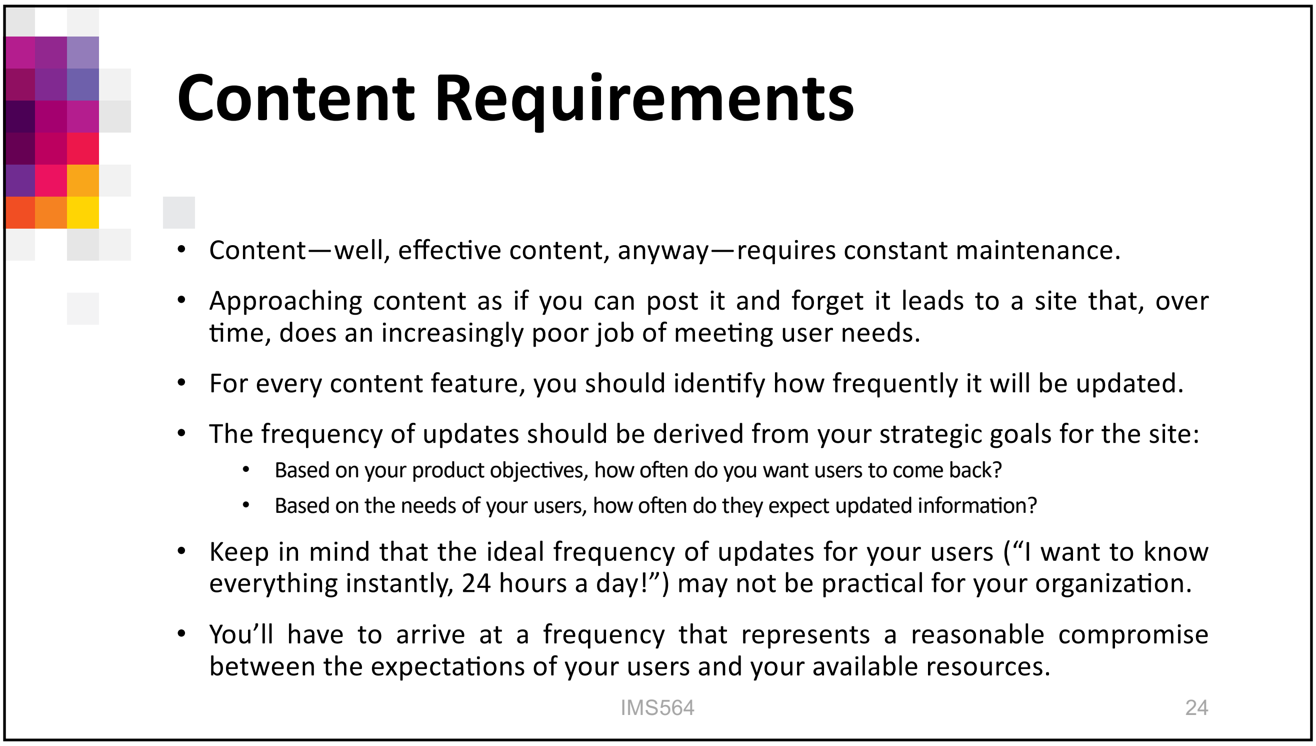
- Content types are
  - text,
  - images,
  - audio, and
  - video
- These different content types can also work together to fulfill a single requirement.
- Identifying all the content types associated with a feature can help you determine what resources will be needed to produce the content (or whether it can be produced at all).

22



Content Requirements

- The expected size of each of your content features has a huge influence on the user experience decisions you will have to make.
- Your content requirements should provide rough estimates of the size of each feature: word count for text features, pixel dimensions for images or video, and file sizes for downloadable, stand-alone content elements like audio files or PDF documents. These size estimates don't have to be precise—approximations are fine.
- Designing a site to provide access to small thumbnail images is different from designing a site to provide access to full-screen photographs; knowing in advance the size of the content elements you have to accommodate enables us to make smart, informed decisions along the way.



Content Requirements

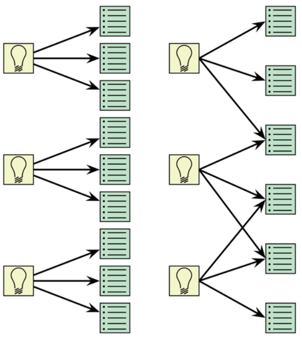
- Content—well, effective content, anyway—requires constant maintenance.
- Approaching content as if you can post it and forget it leads to a site that, over time, does an increasingly poor job of meeting user needs.
- For every content feature, you should identify how frequently it will be updated.
- The frequency of updates should be derived from your strategic goals for the site:
  - Based on your product objectives, how often do you want users to come back?
  - Based on the needs of your users, how often do they expect updated information?
- Keep in mind that the ideal frequency of updates for your users ("I want to know everything instantly, 24 hours a day!") may not be practical for your organization.
- You'll have to arrive at a frequency that represents a reasonable compromise between the expectations of your users and your available resources.

Content Requirements

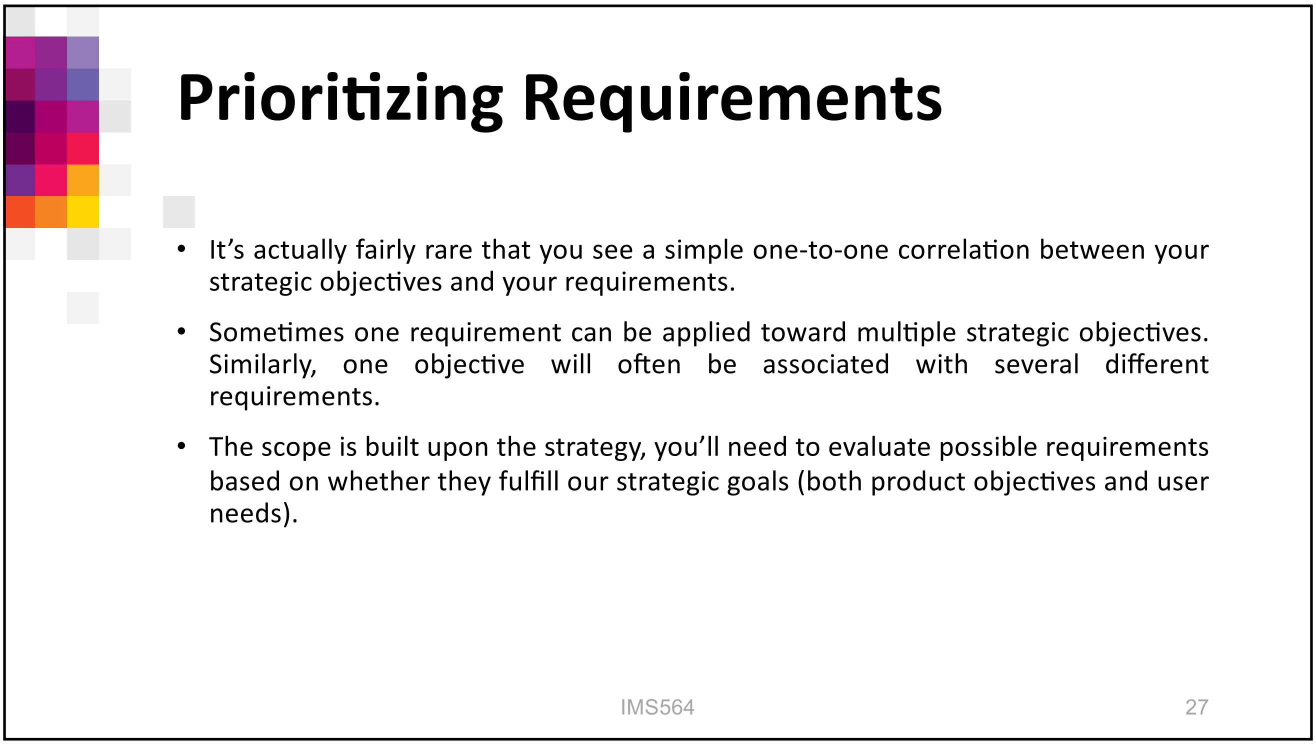
- If your site has to serve multiple audiences with divergent needs, knowing which audience a piece of content is intended for can help you make better decisions about how to present that content.
- Information intended for children requires a different approach from information intended for their parents; information for both of them needs yet a third approach.
- For projects that involve working with a lot of existing content, much of the information that will feed your requirements is recorded in a content inventory.
- By having the inventory (which usually takes the form of a simple, albeit very large, spreadsheet) is important for the same reason that having concrete requirements is important: so everyone on the team knows exactly what they have to work with in creating the user experience.

Prioritizing Requirements

- Everyone who regularly comes in contact with a product—whether they are inside the organization or outside—will have at least one idea for a feature that could be added.
- The tricky part is sorting out what features should be included in the scope for your project.

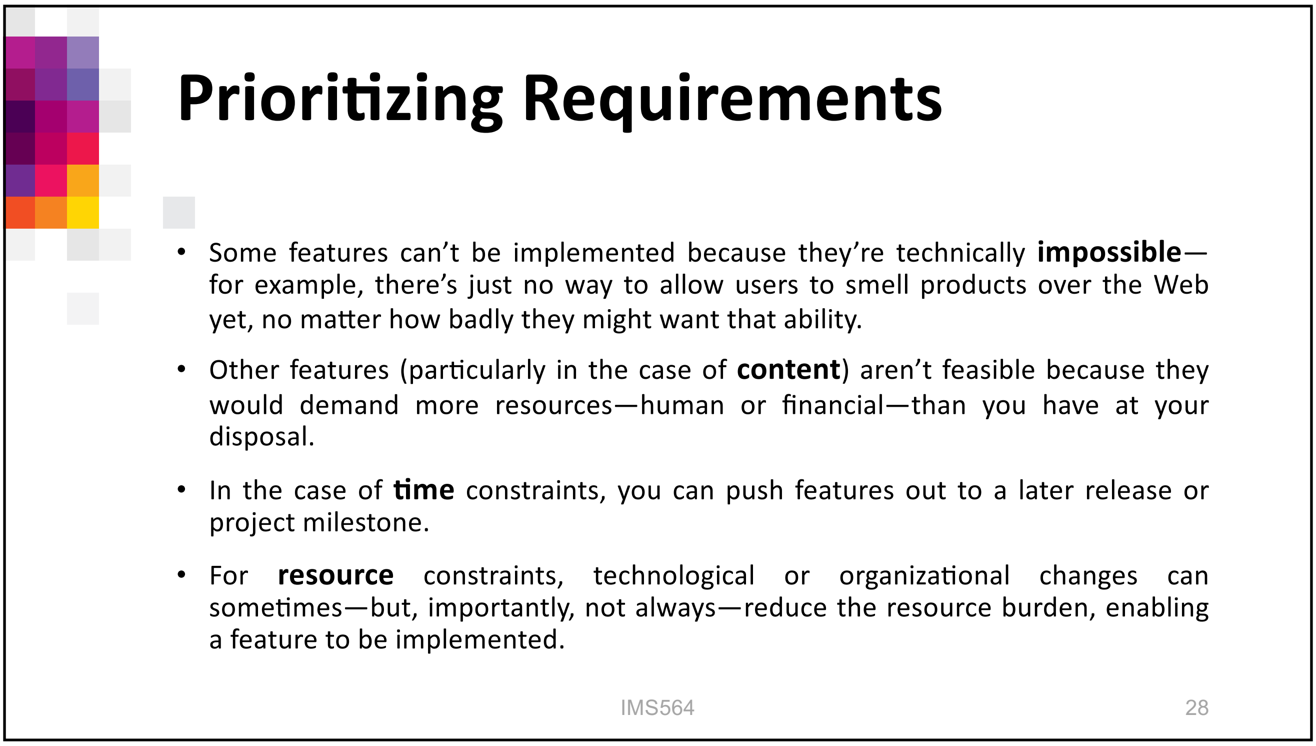


Sometimes a strategic objective will result in multiple requirements (left). In other cases, one requirement can serve multiple strategic objectives (right).



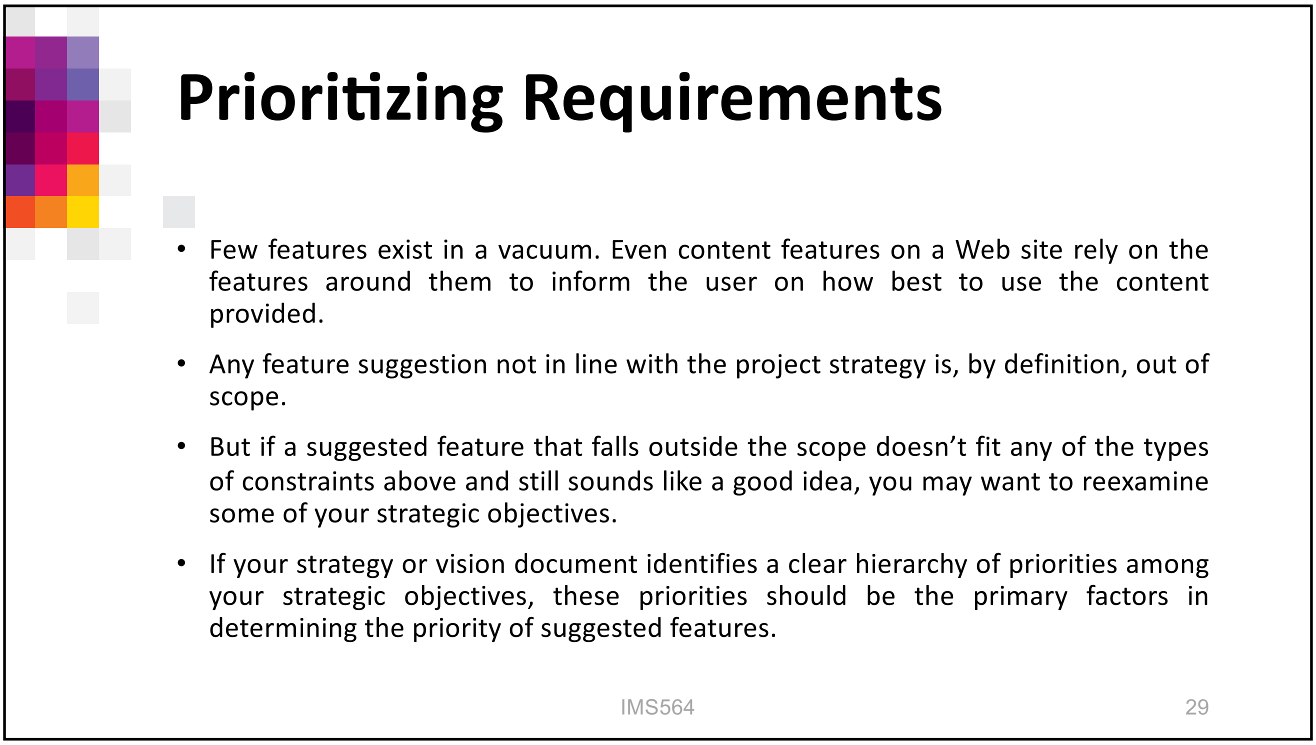
Prioritizing Requirements

- It’s actually fairly rare that you see a simple one-to-one correlation between your strategic objectives and your requirements.
- Sometimes one requirement can be applied toward multiple strategic objectives. Similarly, one objective will often be associated with several different requirements.
- The scope is built upon the strategy, you’ll need to evaluate possible requirements based on whether they fulfill our strategic goals (both product objectives and user needs).



Prioritizing Requirements

- Some features can’t be implemented because they’re technically **impossible**—for example, there’s just no way to allow users to smell products over the Web yet, no matter how badly they might want that ability.
- Other features (particularly in the case of **content**) aren’t feasible because they would demand more resources—human or financial—than you have at your disposal.
- In the case of **time** constraints, you can push features out to a later release or project milestone.
- For **resource** constraints, technological or organizational changes can sometimes—but, importantly, not always—reduce the resource burden, enabling a feature to be implemented.



Prioritizing Requirements

- Few features exist in a vacuum. Even content features on a Web site rely on the features around them to inform the user on how best to use the content provided.
- Any feature suggestion not in line with the project strategy is, by definition, out of scope.
- But if a suggested feature that falls outside the scope doesn’t fit any of the types of constraints above and still sounds like a good idea, you may want to reexamine some of your strategic objectives.
- If your strategy or vision document identifies a clear hierarchy of priorities among your strategic objectives, these priorities should be the primary factors in determining the priority of suggested features.