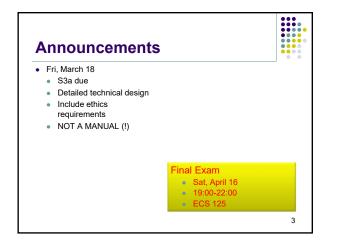


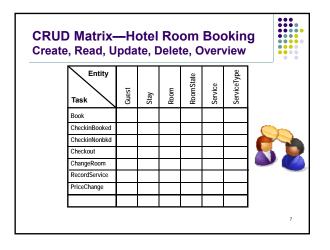
SENG 321 Calendar							
Deliverable C2 (revised)	Thu, Mar 10	C2 feedback on 52a&52b	5% of project				
Quiz 2 Topic: Requirements Engineering Ethics	Fri, Mar 11	In class	3% of course				
	Fri, Mar 18	53a Technical Design Spec	15% of project				
Deliverable S3b	Tue, Mar 22	S3b Manual	10% of project				
Deliverable C3	Thu, Mar 24	C3 feedback on S3a&S3b	10% of project				
Easter break	Mar 25-28	Fri, no class					
Deliverable S4	Mar 29-31	S4 project demo	10% of project				
Deliverable C4	Mar 29-31	C4 feedback on S4	5% of project				
Last Day of Classes	Thu, Mar 31						
Final Exam	Set, Apr 16	19:00-22:00 EC5 125	35%				



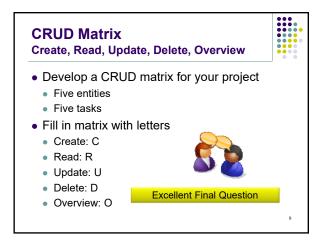
 Key data operate Create 	Operation	SQL	HTTP	DDS
Read	Create	INSERT	PUT / POST	write
 Update 	Read (Retrieve)	SELECT	GET	read / take
 Delete 	Update (Modify)	UPDATE	POST / PUT / PATCH	write
	Delete (Destroy)	DELETE	DELETE	dispose

CRUD: N Inter							
	Customer	Customer Order	Customer Account	Customer Invoice	Vendor Invoice	Product	ו
Receive Customer Order	R	с	CR	2 CO			
Process Customer Order	CRU	<u> </u>	RU		1	R	
Maintain Customer Order	U		U		RU		
Terminate Customer Order	U		U		RU		
Fill Customer Order	RU		RU			RU	1
Ship Customer Order			U		с		1
Validate Vender Invoice				с – СС	R		
Pay Vender Invoice					RU		
Invoice Customer	RU		RU	с			
Maintain Inventory						CRUD	1

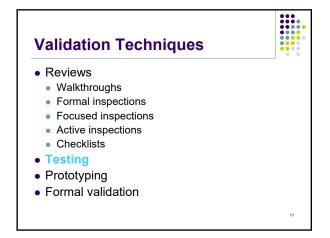


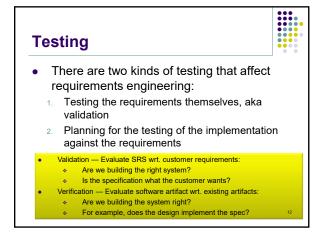


RUD Matrix eate, Read,							g
Entity Task	Guest	Stay	Room	RoomState	Service	ServiceType	
Book	сио	с	0	UΟ			
CheckinBooke	ed RU	UΟ	0	U O			
CheckinNonbl	kd C U O	С	0	UΟ			
Checkout	U	UΟ	R	U			
ChangeRoom	R	R	0	UO			
RecordService	e		0		С	R	
PriceChange			C UDO			C UDO	
	1						



Anothe		וסי			4 mi 1			nla			
ATTOLITE	er CRUD Matrix Example										
	Catalog	Customer	Inventory item	Order	Order item	Order transaction	Package	Product item	Return item	Shipment	Shippe
Look up item availability			R								
Create new order		CRU	RU	С	С	С	R	R		С	R
Update order		RU	RU	RUD	RUD	RUD	R	R		CRUD	R
Look up order status		R		R	R	R				R	R
Record order fulfillment	1				RU					RU	
Record back order					RU					CRU	
Create order return		CRU		RU		С			С		
Provide catalog info	R		R				R	R			
Update customer account		CRUD			-				and the second second		
Distribute promotional package	R	R	R				R	R			
Create customer charge adjustment		RU				CRUD					
Update catalog	RU		R				RU	R			
Create special product promotion	R		R				R	R			
Create new catalog	С		R				CRU	R			1



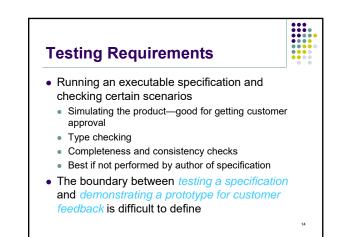


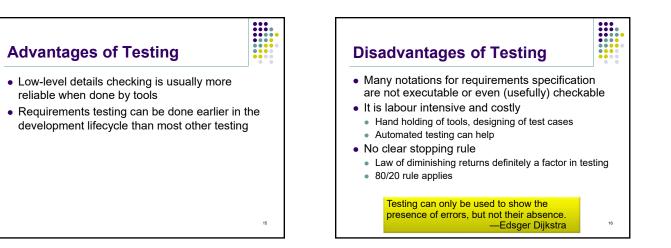
Quantifiable & Non-Quantifiable Requirements

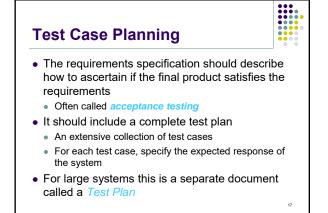
• Quantifiable Requirements

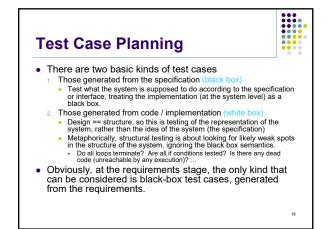
- R: The system m Find a property that provides a scale for measurement within the context (e.g., mins)
- Under what circumstances would the system fail to meet this requirement? Under what circumstances would the system fail to meet this requirement? The stakeholders review the context: failure if a customer has to wait longer than 3 minutes for a response "3 minutes" becomes the quality measure for this requirement

- Non-Quantifiable Requirements
 - R: The automated interfaces of the system must be easy to There is no obvious measurement scale for "easy to learn"
 - Investigate the meaning of the requirement within the particular context, identify limits for measuring the requirement.
 - What is considered a failure to meet this requirement?
 - Novice users: stakeholders want novices to be productive within half an hour
 - Quality measure: a novice must be able to complete a customer order transaction within 30 mins of first using the system
 - S. Robertson. An Early Start to Testing: How to Test Requirements, EuroSTAR '96



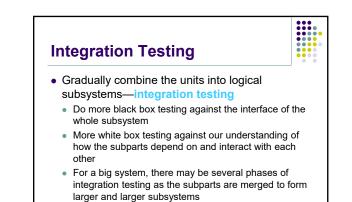






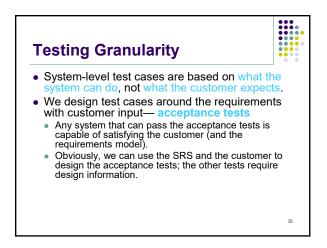
Granularity of Tests

- When we are testing code, we start with unit tests, which are at the level of a class / module / file (depending on the language)
 We try to rigorously test each method / procedure of each unit.
- We try to rigorously test each method / procedure or each unit.
 You should have both black box and white box tests for each unit.
 - The black box tests are designed against the externally visible interfaces of the unit
 - For each method, think of ways of testing it using only your
 - The white box tests are designed against the way in which the code is written
 - For example, try to test all paths through a method, try to exercise all test conditions in ifs and loops, boundary values, etc.



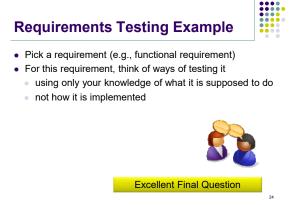
System Testing

- Black box test cases based around what the system as a whole is designed to do
 - Use the top level interface
- White box test cases designed around our understanding of the structure of the design
- It is integration testing at the top level



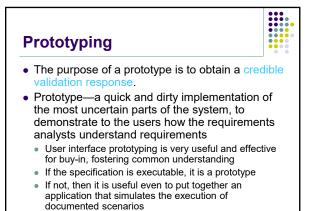
Scenarios as Test Cases Scenarios developed for the purpose of identifying requirements are basically test cases. For example, a scenario gives for each user input the system's response, and lays them out in the order in which they should occur in one computation in the system.

23



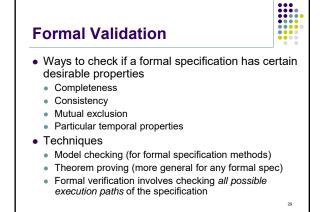
Validation Techniques

- Reviews
 - Walkthroughs
 - Formal inspections
 - Focused inspections
 - Active inspections
 - Checklists
 - Testing
- Prototyping
- Formal validation



Mock-up User Interfaces, Validation Techniques Screens, and Prototypes • Very common and useful Reviews • A picture is worth a thousand words Walkthroughs · Mock-up UIs, screens, and prototypes should not be used Formal inspections before a good understanding of the requirements is reached Focused inspections Customers and users can react quite negatively to a mock-up UI Active inspections Convey the wrong message Checklists Not esthetically pleasing • Use task descriptions instead Testing Much more difficult to disagree with a task than with a UI mock-up Prototyping Customer that these are just suggested screens Formal validation · Establish links between customers and prototype developers and user interface designers

25



5