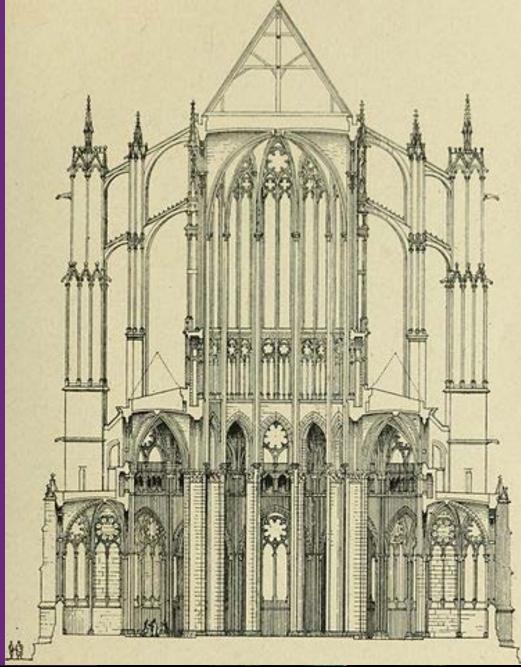
SATURN 2017

13th Software Engineering Institute Architecture Technology User Network Conference

Agile Architecture and Design

Pradyumn Sharma



Title of the Presentation Goes Here © 2017 [Copyright Owner[s]]

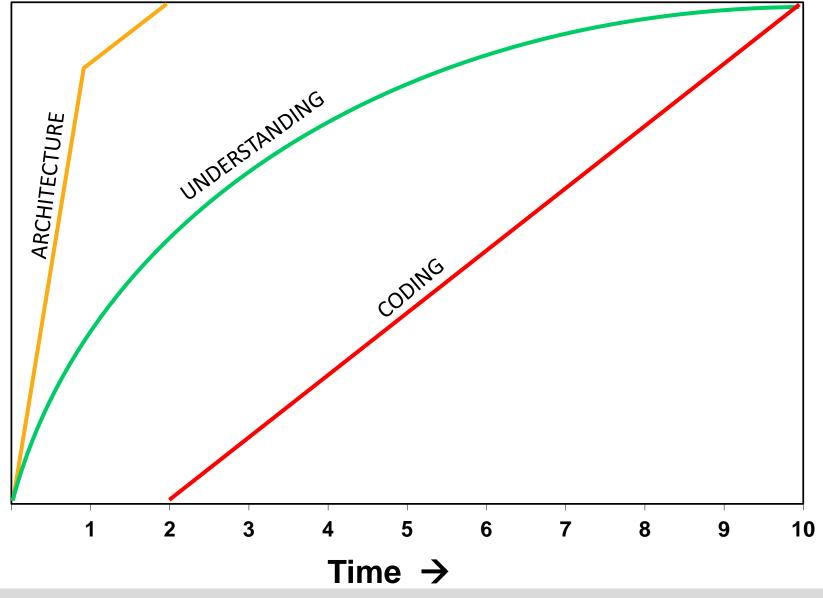
SATURN 2017 Presenter: Pradyumn Sharma



- CEO, Pragati Software, Mumbai, India (www.pragatisoftware.com).
- About 33 years in the IT industry.
- Training and consulting: Agile methodologies, Solution Architecture, Enterprise Architecture, Big Data and Analytics.

SATURN 2017 Architecture: Waterfall





Agile Architecture and Design © 2017 Pradyumn Sharma

Architecture: Agile

- Evolve iteratively
- Through
 - an initial envisioning
 - implementation of stories
 - refactoring and restructuring





Agile Architecture and Design: Overview

- Basics:
 - 1. Involve the entire team
 - 2. Have an "Architecture Owner" role
 - 3. Understand your product
- The key stuff:
 - 4. Create an architecture vision
 - 5. Build architecture through stories
 - 6. Model and implement incrementally



#1. Involve the Entire Team

SATURN 2017 Architecture Role: Traditional

- Characteristics:
 - Specialist
 - Privileged
- Possible consequences (not all may apply):
 - Ivory tower architecture
 - BDUF
 - Analysis paralysis
 - Too abstract
 - Lack of shared understanding among team members



SATURN 2017 Architecture Role: Agile

- Involve the entire team in:
 - Architecture discussion, evolution, implementation
 - Architecture reviews
 - Technical debt sessions
 - Refactoring and restructuring



#2. Have an "Architecture Owner" Role

Title of the Presentation Goes Here © 2017 [Copyright Owner[s]]

SATURN 2017 Architecture Owner

- Master builder
- Depth and breadth of knowledge and skills
- Provides architecture leadership in a collaborative manner



Responsibilities

- DOs:
 - Bring the team together for all discussions regarding architecture envisioning and modeling
 - Facilitate architecture modeling and evolution
 - Help in building a shared understanding
 - Help the team members enhance their capabilities in understanding architectural principles and tradeoffs involved
- DONTs:
 - Dictate the architecture, preventing others from having their say.
 - Guard the architecture as personal property.

#3. Understand Your Product

SATURN 2017 **Understand Your Product**







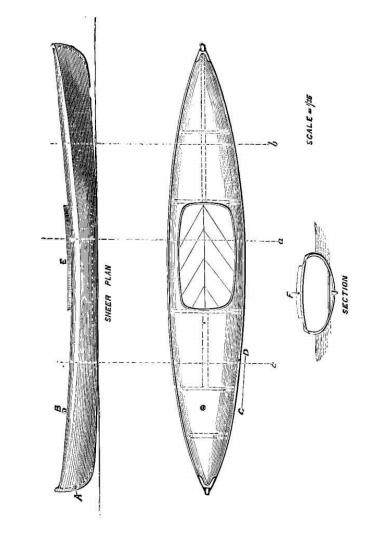




SATURN 2017 Understand Your Product







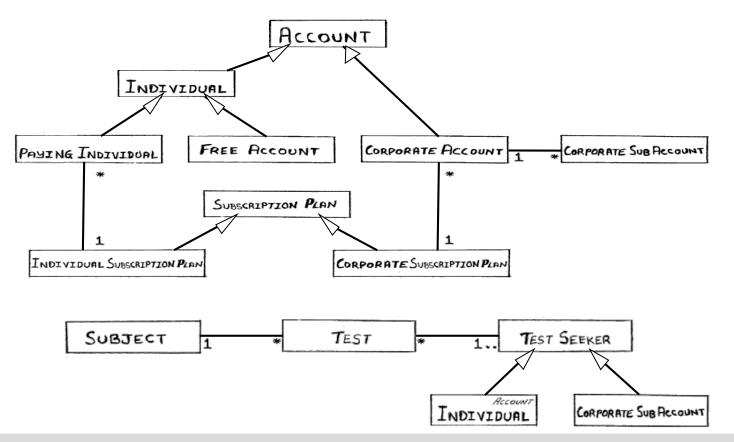
#4. Create an Architecture Vision

SATURN 2017 Create an Architecture Vision

- Create a shared vision of the architecture
- When?
 - Sprint zero.
- How?
 - Architecture / technical workshop.
- Who?
 - Team as well as the Product Owner (architecture must be based on requirements).

Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling





SATURN 2017 Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- **UI** prototyping •

171	EMS				t	Items > New Item
DG	etegory : [More Filters]	Sort by _	يلد	Get date		General Specifications Vendors
ļ	A B C X Y Z Others A	LL		(+ Add no	ero Item)	Iten name
	Showing 1-25 of «First	t <prev.< td=""><td>Next></td><td>Last >></td><td></td><td>Lume I dell' C</td></prev.<>	Next>	Last >>		Lume I dell' C
	Item Uni	t Rate	Stock			Code
	Alfa Black Toner Cartridge N (ABU01)	v. \$ 56.00		Edit	Delet	Description
_	the Americaher AL size 500 sheets	Park \$ 3.99		Edit	Delete	
Ц	(AM 001)					Unit
						-
						Encel Save & Continue Cancel

Save & Exit

Save & Continue

18



Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Identify desired architecture qualities



Architectural Qualities: Examples



- Portability
- Modifiability
- Performance
- Security
- Testability
- Usability
- Availability

- Conceptual integrity
- Accuracy
- Concurrency
- Customization points
- Internationalization
- Operations
- Maintenance
- Environmental impact
- Reliability
- Regulatory compliance
- Serviceability
- Support
- Dependencies on external systems

SATURN 2017 Architectural Qualities: Examples



- Portability
- Modifiability
- Performance
- Security
- Testability
- Usability
- Availability

- Conceptual integrity
- Accuracy
- Concurrency
- Customization points
- Internationalization
- Operations
- Maintenance
- Environmental impact
- Reliability
- Regulatory compliance
- Serviceability
- Support
- Dependencies on external systems

Search Engine:

- •Performance
- •Availability
- •Portability

Stock Trading System:

- •Security
- •Dependencies on external
- systems
- •Performance
- **Online Travel Agency:**
- •Dependencies on external
- systems
- •Usability
- •Internationalization

SATURN 2017 Architecture Requirements

- Be specific
- What exactly do we mean by an architecture requirement, in a given context?

Portability Requirements: Examples

- System should run on all popular web browsers as well as smartphones.
- It should be easy to migrate data from one database platform to another.
- Customers should be able to choose the database platform for their respective deployments.

Security Requirements: Examples

- Preventing unauthorized access to data or services
- Dealing with DoS attacks
- Non-repudiation (a transaction cannot be denied by any party)
- Secure transmission of sensitive data



Usability Requirements: Examples

- Ability to save incomplete data as draft and resume later
- Appropriate feedback to the users about state of completeness of a business process / long-running transaction / multi-step data entry, etc.
- Ability to bookmark, roll back, etc.

Availability Requirements: Examples



- System uptime should be at least 99.9%
- When a system service fails, an alert to the Infrastructure team should be raised.
- Even when one service of the system becomes unavailable the other services should continue to run as far as possible.
- It should be possible to deploy newer versions of the components without shutting down the system.

Integrity Requirements: Examples

- System should detect any issues with data consistency or integrity that creep in.
- Likelihood of concurrent access to data: very high / medium / low. => Need for concurrent edits without compromising data integrity.

Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Identify desired architecture qualities
- Strategies for achieving desired qualities; don't commit

Portability: Strategies

- Separation of UI logic, business logic, data access logic in separte components
- Layering, partitioning

Modifiability: Strategies

- Loose coupling between components
- Proper encapsulation
- Layering, partitioning
- Model-View-Controller
- Application framework providing common features and services, extention points for providing story-specific behaviour
- OO principles and design patterns
- Anticipating changes, providing hooks to facilitate changing behaviour
- Non-architectural aspects, such as coding conventions and techniques

SATURN 2017

Concurrency: Strategies

- Use of Transaction Managers
- Optimistic locking / Pessimistic locking.
- Fine-grained locking / Coarse-grained locking

Usability: Strategies



- Separation of the UI from the rest of the application
- Giving feedback about what the system is doing
- Letting the user issue commands such as Save as Draft, Cancel, Undo, show multiple views
- Design patterns: Command, Memento
- Maintaining a model of the task, or the system, or the user

SATURN 2017 Performance: Strategies

- Infrastructure planning: server capacity, clustering, failover, virtualization, network bandwidth
- Managing event rate
- Quantum of communication among components, layering
- Database: indexes, partitions, stored procedures
- Non-architectural aspects:
- choice of algorithms
- implementation of selected algorithms
- writing efficient database queries

SATURN 2017

Security: Strategies

- Authentication of users
- Single sign-on
- Authorization of users, limiting access
- Audit trail
- Intrusion detection system



Integrity: Strategies



- Periodic run of batch programs to check integrity of derived data against raw data
- Discover patterns, fix integrity problems automatically, maintain history of such changes

Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Desired architecture qualities
- Strategies for achieving desired qualities; don't commit
- Cross-cutting requirements



Cross-Cutting Requirements: Examples

- Audit trail
- Alerts for important events that need attention
- Centralized error logging
- Excel export from all browse windows

Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Desired architecture qualities
- Strategies for achieving desired qualities; don't commit
- Cross-cutting requirements
- Other considerations for architecture



SATURN 2017 Other Considerations: Examples

- Target market
- Time to market
- Rollout schedule
- Projected lifetime of the system
- Cost and benefit
- Correctness and completeness



Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Desired architecture qualities
- Strategies for achieving desired qualities; don't commit
- Cross-cutting requirements
- Other considerations for architecture
- Validate (at every stage) with the customer



Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Desired architecture qualities
- Strategies for achieving desired qualities; don't commit
- Cross-cutting requirements
- Other considerations for architecture
- Validate (at every stage) with the customer
- Potential technical risks



Potential Technical Risks: Examples

- Technology being used is not mature
- Product is very complex to implement
- Integration requirements are very complex and diverse
- Requirements are unclear, unstable

Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Desired architecture qualities
- Strategies for achieving desired qualities; don't commit
- Cross-cutting requirements
- Other considerations for architecture
- Validate (at every stage) with the customer
- Potential technical risks
- Take stock of existing, reusable architectural assets



SATURN 2017 Architecture Workshop: Activities

- Using whiteboard / paper...
- High level domain modeling
- UI prototyping
- Desired architecture qualities
- Strategies for achieving desired qualities; don't commit
- Cross-cutting requirements
- Other considerations for architecture
- Validate (at every stage) with the customer
- Potential technical risks
- Take stock of existing, reusable architectural assets
- Prioritize the architecture features. Based on:
 - Business value
 - Cost of implementing early vs cost of implementing late



Architectural Features' Prioritization: Example



- Separation of UI, database, business logic, workflow logic into separate, distributed layers. (Benefits: portability, modifiability).
- Creation of an application framework. (Benefits: avoiding duplication of code in various stories; consistency of behavior; ease of adding cross-cutting requirements).
- Pessimistic Locking for concurrency management.
- Usability features in the framework: Save as Draft, Undo, Redo.
- Cross-cutting requirement: Audit trail.

Product Backlog



Produ	ct Backlog
Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to extract common behaviour into a "framework" for the application
5	Testing the framework by extending it to implement a third story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and testing
7	Account Heads: for "Customer" types, entering additional details
8	Framework: Including support for Save as Draft, Undo, Redo.
9	Framework: Audit Trail.

#5. Build Architecture Through Stories

Title of the Presentation Goes Here © 2017 [Copyright Owner[s]]

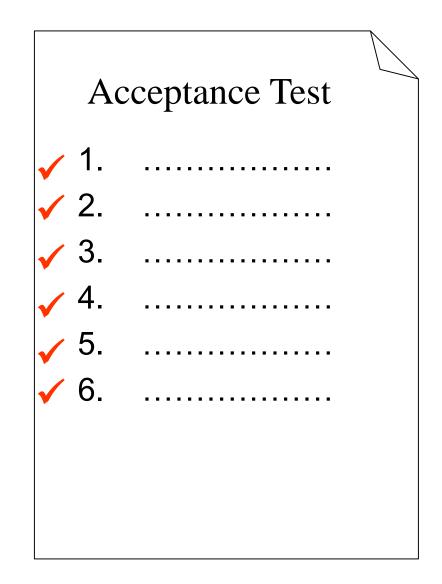
SATURN 2017 Product Backlog



Produ	ct Backlog
Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to
	extract common behaviour into a "framework" for the
	application
5	Testing the framework by extending it to implement a third
	story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and
	testing
7	Account Heads: for "Customer" types, entering additional
	details
8	Framework: Including support for Save as Draft, Undo,
	Redo.
9	Framework: Audit Trail.

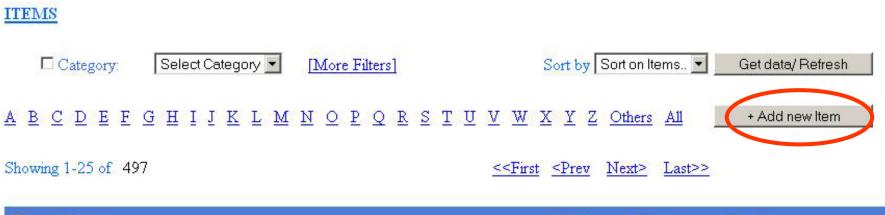
Saturn 2017 Story 1: Creating List of Items





Implement the UI and the Story





Items	Unit	Rate	Stock	
Alfa Black Toner Cartridge (AB001)	No.	\$56.00		<u>Edit</u> <u>Delete</u>
AmeriPaper A4 size 500 sheets (AM001)	Pack	\$3.99	31111	<u>Edit</u> <u>Delete</u>

Form_load: fetch data from Items table populate the grid New_click: display blank entry form

Implement the UI and the Story

General	Specifications	Vendors		
Item		[
Name		<u>.</u>		
Code				
Descrip	tion		×	
Unit		Select Unit	•	
Coup	& Exit S	ave & Continue	Cancel	

SaveAndExit_click: validate data INSERT into Items values ... exit form etc.



Implement the UI and the Story



Form_load:

fetch data from Items table

populate the grid

New_click:

display blank entry form

//...on return from the entry form

re-populate the grid from Items table

etc.

Product Backlog

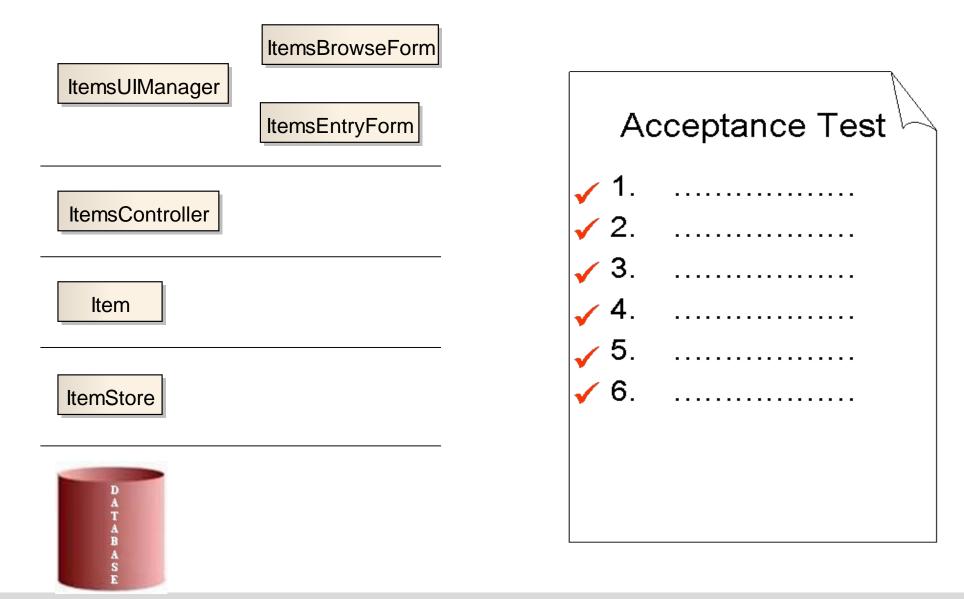


Product Backlog

Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to extract common behaviour into a "framework" for the application
5	Testing the framework by extending it to implement a third story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and testing
7	Account Heads: for "Customer" types, entering additional details
8	Framework: Including support for Save as Draft, Undo, Redo.
9	Framework: Audit Trail.

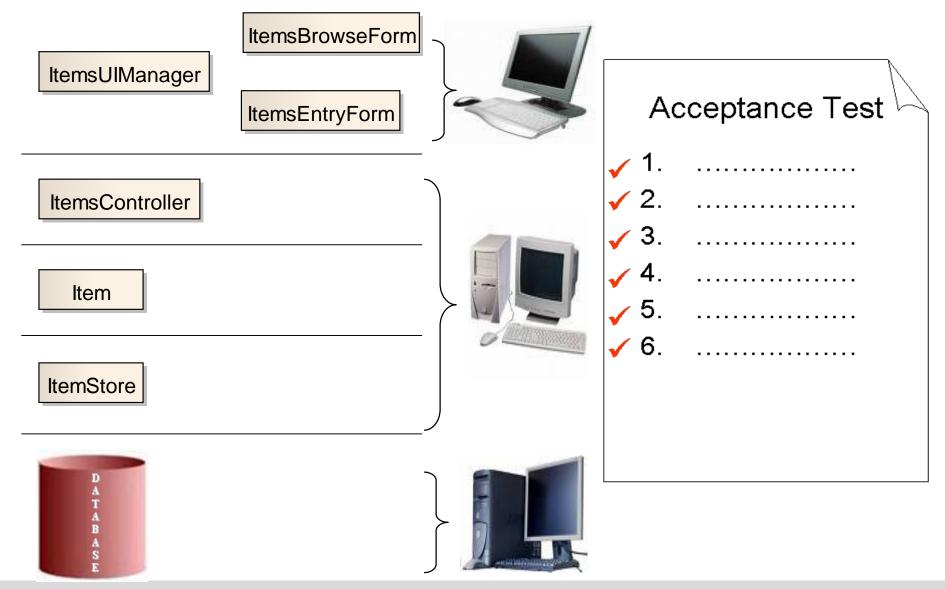
Story 2: Layered Architecture





SATURN 2017 Distributed Architecture





Looking Back



- Miniscule architecture evolved, proven with the code
- Architectural stability and state of completeness: very low



Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to extract common behaviour into a "framework" for the
	application
5	Testing the framework by extending it to implement a third story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and testing
7	Account Heads: for "Customer" types, entering additional details
8	Framework: Including support for Save as Draft, Undo,
	Redo.
9	Framework: Audit Trail.

Story 3: Creating List of "Account Heads"

- Write acceptance tests.
- Design the UI.
- Start implementing the story.
- Similarity in implementation with the earlier story...
 => take up the next story also

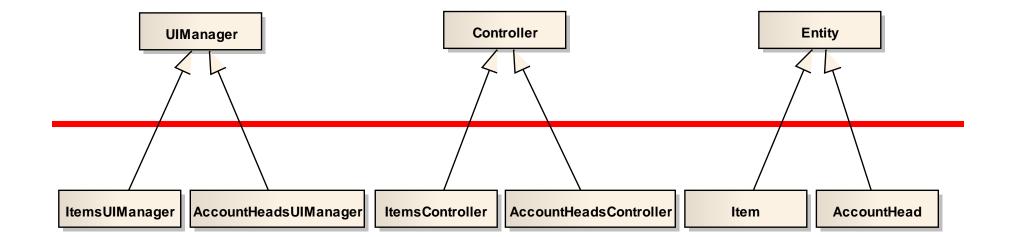


Product Backlog



Produ	ct Backlog
Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to
	extract common behaviour into a "framework" for the
	application
5	Testing the framework by extending it to implement a third
	story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and
	testing
7	Account Heads: for "Customer" types, entering additional
	details
8	Framework: Including support for Save as Draft, Undo,
	Redo.
9	Framework: Audit Trail.

Saturn 2017 Story 4: Application Framework Evolution



Product Backlog



Product Backlog

Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to extract common behaviour into a "framework" for the application
5	Testing the framework by extending it to implement a third story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and testing
7	Account Heads: for "Customer" types, entering additional details
8	Framework: Including support for Save as Draft, Undo, Redo.
9	Framework: Audit Trail.

Story 5: Test Framework, "Tax Codes" Story

- Implement a third story, by extending the framework.
- Helps in smoothening some rough edges in the framework.





Produ	ct Backlog
Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to
	extract common behaviour into a "framework" for the
	application
5	Testing the framework by extending it to implement a third
	story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and
	testing
7	Account Heads: for "Customer" types, entering additional
	details
8	Framework: Including support for Save as Draft, Undo,
	Redo.
9	Framework: Audit Trail.

Story 6: Framework, Handling Concurrency

- Implementing database record locking in the framework, instead of in individual stories.
- Locking strategies:
 - Pessimistic locking
 - Optimistic locking
 - No locking
- Who chooses the locking strategy?
- And how do they choose it?
- How do we help them make a choice?
- By writing scenarios with UI prototyping.



Produ	ct Backlog
Sr.No.	Story
1	Maintaining the list of "Items" for stock and sale
2	Implementing Layering and Partitioning in "Items" story
3	Creating the list of "Account Heads"
4	Refactoring the "Items" and "Account Heads" stories to
	extract common behaviour into a "framework" for the
	application
5	Testing the framework by extending it to implement a third
	story (Tax Codes), and ironing out issues.
6	Implementing concurrency handling in the framework and
	testing
7	Account Heads: for "Customer" types, entering additional
	details
8	Framework: Including support for Save as Draft, Undo,
	Redo.
9	Framework: Audit Trail.

Saturn 2017 Similarly...



- Story 7: Account Heads, entering additional details for "Customer" type
- Story 8: Framework, support for Save as Draft, Undo, Redo
- Story 9: Framework, Audit Trail
- Likewise for other cross cutting functional or UI requirements. Examples:
 - Change history
 - Pagination in browse windows
 - Search within a browse window
 - Excel export from all browse windows

#6. Model and Implement Incrementally

Model and Implement Incrementally

- Model throughout the lifecycle, in small increments.
- Split large, complex stories.
- Example: Trainer tracking chart in a Training Monitoring System.

Example: Trainer Tracking Chart



February	, 2008	Prev	vious	Next	Refrest	n Expe	ort Cor	nfirm ₩ 0	New	Confirm 🔤	▼ Edit	▼ E	dit History	Exit
Trainer Track	ing Chart	For : Febr	ruary, 2008	8 [Total]	rainings =	269.25 (li	nternal) + 1	100.75 (Fre	eelancer) =	= 370 day(s)]			
Trainer(s)	8-Sat	10-Sun	11-Mon	12-Tue	13-Wed	14-Thu	15-Fri	16-Sat	17-Sun	18-Mon	19-Tue	20-Wed	21-Thu	22-Fri
Vipul	Citigro									#Intelen				#Heinz I
Foram	Tata S		+Relianc	+[2] Kirlo	+[2] Kirlo	Leave.	+Tata S	+Tata S			Leave .	+Reliance	+Reliance	Leave.
Varsha	Relianc			#lflex So	#lflex So	#lflex So	#lflex So			+Relianc	+Relianc		+Lehma	+Lehma
Viraj	Godrej			#Abbott I	#Abbott I		#Accel F	#Bharti			#Transoc	#Clariant	#Clariant	+Dun &
Swati	[2] Reli		+Mahind	Leave	Leave		#Nelco L	#Nelco L				Leave		
Anuradha	Dewan		#The Gre	#The Gre	Leave.	Attendin	Attendin		#Bank of			+MRC Tra		
Mrinal	1		Attendin							AttendTr	AttendTr			
Rajesh	Larsen	Leave.		Leave.	Prep On	Prep On	+Bharti							+Mazag
Altaf	SGVe	\$SGVe			\$Syntel I	\$Syntel I	\$Syntel I			+ICICI Pru	+ICICI Pro	+The Cre		+ICICI P
Manisha			#[2] Lehm	#[2] Lehr		+[2] Leh	+[2] Leh	#khaitan		#lflex So	#lflex So	#lflex So	#Panora	
Zubair	hternal t			Travel	\$Godrej	\$Godrej	Travel			#Transoc			+Godrej	+Godrej
RajVaidya –	AFL Pri		Leave	Leave	Leave	Leave	Leave	Leave		Leave.		#Transoc		#Transo
Amardeep	Videsh		+Siemen	+Siemen			#Merck S	#Merck S		#Hindust	#Hindust	#Hindust	-	
SujeetK	Mercat		#Hindust	#Hindust	#Hindust	Leave	Travel.	\$ICICI P	\$ICICI P	Travel.	+[2] Pru	+[2] Pru	+Viacom	
Chandra	eave.		Evaluato	Evaluato	+[2] Leh	+[2] Leh	+[2] Leh	+[2] Leh		Attend "	Attendin	Attendin	Attendin	Attendin
Bhavna	eave.		#Datam	#Datam	#Datam	#Datam	#Datam	#Datam		#Datam	#Datam	#Datam	#Datam	#Datam
Shraddha			Cert On	Attendin						+Bave T	+Bave T	+Bave T		+Rave T
Avnika	alf Day		Leave.				Work ord	er Number	C/2007/	2095 - S			Day1 of 3	
Abhishek			Leave.	Attendin	Attend L	Attend L	Venue		Client loc	ation				+[2] Leh
Anushka				Cert On	Attend L	Attend L	Location		MIDC , K	Inowledge	Center			Attend T
Ankur			Part for				Client		Datamat	ics Limited	- 15			
Vishal	Larsen	\$Larsen	\$Larsen	Unavaila	Unavaila	Unavaila	Subject		Visual Ba	asic .NET	2.0			Unavaila
Merlin	Larsen		#Larsen	#Larsen		Prep For	Timinas		09:30 AN	/ to 05:30	PM			Half Day
Minal	eave.	Leave.	Leave.	Attendin	Leave.		ME		Rosy					Attend T
Aruna		+[2] Gau					Expected	Paxi	15	Actual Pa	av 1	2		
Pradyumn								\$Symant	10	Hotadirit		+[2] PXL	+[2] PXL	+[2] PXL
Majrul	DC Off			\$Mastek	\$Mastek	#Mastek	#Mastek			#Mphasis		#Iflex So		
Yatin Naik														
Chetan														
Ganesh B	1		\$Capge	\$Capge	\$Capge									
Girish Rao	Larsen		#Larsen	#Larsen			#Larsen 8	#Larsen 8		#Larsen 8	#Larsen 8	#Larsen 8	#Larsen 8	
Chitra	Rave T							+Rave T	+Mahes				+CFC In	+CFC In
Vaishali										+Lodha	+Lodha	+Lodba		

Model and Implement Incrementally



		February	, 2008	Pre	vious	Next	Refres	h Exp	ort Co	nfirm W O	New	Confirm [- Edit	E	dit History	Exit	
•	Static grid wi	th ha	ard	For - Feb	de	dd	ata	269.S	im	ble	nav	vid	atic	n k	(ev	Saeri	
	-	Vipul	Citigro									#Intelen			l ju	#Heinz In	
•	Dynamic gric	Foram Varsha	Tata S Refere	r∩tr	+Relianc	+[2] Kirlo	+[2] Kirlo	Leave.	+Tata S	Tata S	22	+Relianc	Leave . +Relianc	+Reliance	e +Reliance +Lehma	Leave. +Lehma	
-	Dynamic grie	Viraj	Godrej				#Abbott				5 C.	Trollario		#Clariant	#Clariant		
•	Cacha tabla	Swati Afradha 4	[2] Reli Dewan	-fra	+Mahind #The Gr		Leave	Attendio		. #Nelco L	#Bank of		NIO	Leave +MPC T:	atio		
•	Cache table.	Auto			esn					5 111	ce	IS:	I.tt.N.O	vig	all		
	months.	Altaf	Larsen S G Ve	Leave. \$S G Ve		Leave.	Prep Un \$Syntel I	Prep On \$Syntel I	+Bharti \$Syntel I			+ICICI Pri	u +ICICI Pri	u +The Cre	:0	+Mazag +ICICI Pru	
	monuis.	Manisha			#[2] Lehr	n #[2] Lehr			+[2] Leh	#khaitan				#Iflex So			
	_	Zubair	hternal t			Travel	\$Godrej	\$Godrej	Travel			#Transoc			+Godrej	+Godrej	
\bullet	Cut, Copy, Pa	aste	KAFLETTI. Midush	XCE		xnc	brf (nnt		Leave	:	Leave. #Hindust	#Hindust	#Transoc #Hindust	= #Transoc +[2] Tata		
		SujeetK	Mercat	ΛΟ	#Hindust	: #Hindust	#Hindust	Leave	Travel.	SICICI P	\$ICICI P	Travel.	+[2] Pru	+[2] Pru	+Viacom		
		Chandra	eave.		Evaluato	Evaluato	+[2] Leh	+[2] Leh	+[2] Leh	+[2] Leh				Attendin	Attendin	Attendin	
lacksquare	Options to ac		edi	t w	ork	Or	der	Satr	om	#Datim	e a	rid.	#Datam	#Datam	#Datam	#Datam	
		Avnika	lalf Day		Leave	Attendin		U	Work ord	ler Number	C/2007/	2095 - S	L+Bave L	1+Bave L	L+Rave T Day 1 of 3	+Have I	
	— 1 · 7 ·	Abhishek		-	Leave.	Attendin	Attend L	Attend L	Venue		Client loo	ation				+[2] Leh	
•	Tracking of ir	ncrei	me	nta		nar	ICIE	Stend	Lhe	and	en	ISt	DrV.	i i		Attend T	
	3	Ankur	<u> </u>		Part for		U		Client	3	Datamat	ios Limiteu	10				
		Vishal Merlin	Larsen Larsen	\$Larsen		Unavaila #Larsen	Unavaila	Unavaila Prep For	Subject			asic .NET 4 to 05:30				Unavaila Half Day	
		Minal	eave.	Leave.	Leave.		Leave.	Гіергоі	Timings M E		Rosy	4 (0.00:30	IT IM			Attend T	
		Aruna		+[2] Gau					Expected	d Pax.:	15	Actual P	ax.: 1	12			
		Pradyumn								\$Symant				+[2] PXL	+[2] PXL :	+[2] PXL :	
		Majrul	DC Off			\$Mastek	\$Mastek	#Mastek	#Mastek			#Mphasis	s #Mphasis	#Iflex So	#Iflex So	#Iflex So	
		Yatin Naik Chetan															
		Ganesh B			\$Capge	\$Capge	\$Capge										
		Girish Rao	Larsen			#Larsen			#Larsen	8 #Larsen 8	5	#Larsen (8 #Larsen 8	8 #Larsen 8	8 #Larsen 8	S	
		Chitra	Rave T							+Rave T	+Mahes				+CFC In	+CFC In	
		Vaishali										+Lodha	+Lodha	+Lodha			

Model and Implement Incrementally



- At the start of each iteration, during the sprint planning meeting, have discussions on incremental modeling, design changes.
- Apply architecture and design patterns as required, gently.
- Don't worry about getting your architecture right on the first day.
- Test-driven development.
- Refactoring.
- Build automation.
- Continuous integration.

To Summarize

- Evolve architecture, collaboratively
- Establish a clear architecture vision
- Just enough architecture:
 - Identify and fulfill the architecture requirements
 - Establish and validate the architecture with the help of stories
 - Understand the significant elements and how they fit together
 - Identify and mitigate the key risks





13th Software Engineering Institute Architecture Technology User Network Conference

Thank You

Pradyumn Sharma pradyumn.sharma@pragatisoftware.com Twitter: PradyumnSharma



Title of the Presentation Goes Here © 2017 [Copyright Owner[s]]