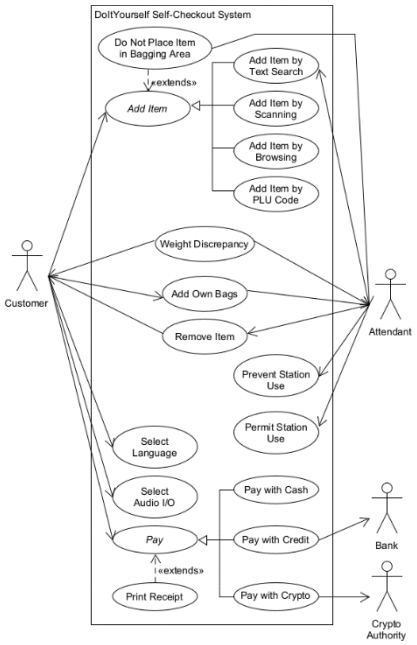
DoItYourself Self-Checkout System Use Cases (v1.0)



The use case diagram to the left represents the overall system, software and hardware. The actors that interact with the system are therefore the Customer and Attendant (humans), the Bank (which can be whichever financial institution), and some sort of "Crypto Authority" (whose details are rather vague).

This diagram is important for the hardware department to understand what hardware is needed to interact with the external context and to support the use cases.

The second use case diagram is drawn from the perspective of just the software portion of the self-checkout system. Therefore, the actors are portions of the hardware plus the "Crypto Authority", and the Customer and Attendant have gone away. Customer I/O and Attendant I/O each represents a combination of touch screen, speaker, and microphone.

The use case descriptions to follow will be relative to the software.

Use Case Descriptions

Use case:	*Add Own Bags*
Primary actor:	Customer I/O.
Goal in context:	To allow the customer to add their own bags to the bagging area without causing a weight discrepancy.
Preconditions:	The system is ready to detect weight discrepancies.
Trigger:	The customer decides to make use of their own bags, activating an appropriate control on their I/O.
Scenario: 1. Customer I/O: Signals that the customer wants to add their own bags. 2. System: Indicates that the customer should add their own bags now. 3. Customer I/O: Signals that the customer has finished adding their own bags. 4. Bagging Area: Signals to the System the weight change. 5. System: Blocks the self-checkout station from further customer actions. 6. System: Signals to the Attendant I/O the need to approve the added bags. 7. Attendant I/O: Signals approval of the added bags. 8. System: Unblocks the self-checkout station. 9. System: Signals to the Customer I/O that the customer may now continue.	
Exceptions: 1. The System is not ready to note weight discrepancies. 2. The attendant does not want to approve the added bags. Presumably, the attendant is aware that the customer has done something improper and communicates directly with customer about the issue. The attendant should thus be able to cancel the request once the customer eliminates the weight discrepancy.	
Priority:	Low, can be skipped without major impact on system operation.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired connection.
Secondary actors:	Bagging Area, Attendant I/O.
Channels to secondary actors: Bagging Area: Hardwired connection. Attendant I/O: Local area network.	
Open issues: 1. It is not clear why the System might not be ready to note weight discrepancies. Once it has completed its startup, each station ought to be ready for customer use.	

Use case:	*Remove Item*
Primary actor:	Attendant I/O.
Goal in context:	To remove an item from a customer's bill, allowing them to also remove it from the bagging area without causing a weight discrepancy.
Preconditions:	The system is ready to detect weight discrepancies.
Trigger:	The customer wishes to eliminate an item from their bill, communicating this to the attendant.
Scenario: 1. Attendant I/O: Signals that a specific item is to be removed from the customer's bill. 2. System: Blocks the self-checkout station from further customer actions.	

3. System: Removes the item from the customer's bill, reducing the expected weight in the bagging area. 4. System: Signals to the Attendant I/O that the bill removal was successful and that the item should be removed from the bagging area. 5. System: Signals to the Customer I/O that the item should be removed from the bagging area. 6. System: Unblocks the self-checkout station.	
Exceptions: 1. The System is not ready to note weight discrepancies. 2. The item is not removed from the bagging area; see **Weight Discrepancy**.	
Priority:	Medium, necessary for good customer service and to avoid having to cancel transactions entirely.
When available:	Third iteration
Frequency of use:	A few times per day.
Channel to actor:	Local area network.
Secondary actors:	Bagging Area, Customer I/O.
Channels to secondary actors: Bagging Area: hardwired, physical connection Customer I/O: local area network; physical connection	
Open issues: 1. Should the request be made verbally to the attendant? What if the customer and attendant do not speak the same language? Should the customer be able to indicate the item on their bill that they want to remove, with the attendant merely approving it and checking that the item be removed from the bagging area?	

Use case:	*Add Item*
Primary actor:	Customer I/O
Goal in context:	To allow the customer to add an item to their bill for purchase.
Preconditions:	The system is ready to detect weight discrepancies and to take customer input.
Trigger:	The customer wishes to add an item to their bill.
Scenario: 1. (Abstract use case) Details about the item to add must be provided to the System. 2. System: Blocks the self-checkout station from further customer interaction. 3. System: The expected weight in the Bagging Area is updated. 4. System: Signals to the Customer I/O to place the item in the Bagging Area. 5. Bagging Area: Signals the weight change from the added item. 6. System: Unblocks the station.	
Exceptions: 1. The customer fails to place the item in the Bagging Area; see **Weight Discrepancy**.	
Priority:	High, the system cannot function with this.
When available:	First iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Hardwired connection.
Secondary actors:	Bagging Area.
Channels to secondary actors: Hardwired connection.	
Open issues:	

The manner in which the customer can indicate what is to be added is unclear. It seems reasonable that barcode scanning would be the default and so the system needs to be ready to scan each item unless an alternative means for providing the information is explicitly selected.

Use case:	*Add Item by Scanning*
Primary actor:	Laser Scanner
Goal in context:	To permit the customer to scan a barcode of an item to add it to their bill for purchase.
Preconditions:	The system is ready to accept customer input.
Trigger:	The customer wishes to scan a barcode on an item that possesses one.
Scenario: 1. Laser Scanner: Detects a barcode and signals this to the System. 2. System: Blocks the self-checkout station from further customer interaction. 3. System: Determines the characteristics (weight and cost) of the product associated with the barcode. 4. System: Updates the expected weight from the Bagging Area. 5. System: Signals to the Customer I/O to place the scanned item in the Bagging Area. 6. Bagging Area: Signals to the System that the weight has changed. 7. System: Unblocks the station.	
Exceptions: 1. The weight in the Bagging Area does not correspond to expectations; see **Weight Discrepancy**. 2. An item is scanned when a customer session is not in progress. The scanned information shall simply be ignored.	
Priority:	High, must be implemented.
When available:	Iteration 1.
Frequency of use:	Every customer transaction.
Channel to actor:	Hardwired connection.
Secondary actors:	Bagging Area, Customer I/O.
Channels to secondary actors: Bagging Area: Hardwired connection. Customer I/O: Hardwired connection.	
Open issues: None.	

Use case:	*Add Item by Text Search*
Primary actor:	Attendant I/O.
Goal in context:	To permit the attendant to add an item to the customer's bill for purchase, knowing details of the item that are not available to the customer (such as the name used by the organization to describe the item's product). This is important if a barcode or PLU code is absent or not recorded within the system.
Preconditions:	The system is ready for attendant input and a customer session is in progress at a connected self-checkout station.
Trigger:	The customer has asked the assistance of the attendant in adding an item to their bill for purchase.
Scenario:	

1. Attendant I/O: The station is selected on which the relevant customer has a session. 2. Attendant I/O: Permits the attendant to search for a product by specifying some text; this should be used as a keyword search in case a product's name is in a different order than expected by the attendant. 3. Attendant I/O: Displays the results of the search. If the results are satisfactory for the attendant, proceed to 3, else return to 1. 4. Attendant I/O: A product in the displayed results is selected for addition. 5. System: Blocks the self-checkout station from further customer interaction. 6. System: Adds the product to the customer's bill, updates the expected weight. 7. System: Signals to the Customer I/O that the item should be added to the Bagging Area. 8. Bagging Area: Signals to the System that the weight has changed. 9. System: Unblocks the station.	
Exceptions: 1. The weight in the Bagging Area does not correspond to expectations; see **Weight Discrepancy**.	
Priority:	Medium, should be implemented as the system will not function well without it.
When available:	Third iteration.
Frequency of use:	A few times per day.
Channel to actor:	Hardwired.
Secondary actors:	Bagging Area, Customer I/O.
Channels to secondary actors: Bagging Area: Local area network. Customer I/O: Local area network.	
Open issues: 1. Text search is impractical without a physical keyboard. Presumably the attendant I/O will have to possess a physical keyboard but not the individual self-checkout stations.	

Use case:	*Add Item by Browsing*
Primary actor:	Customer I/O.
Goal in context:	To allow the customer to look through a visual catalogue of products to select the one for which they wish to add an item for purchase.
Preconditions:	The system is ready for customer input.
Trigger:	The customer does not have a PLU code or barcode available for an item, but they want to add the item to their purchase.
Scenario: 1. Customer I/O: Displays the visual catalogue, allowing the customer to browse through it. 2. Customer I/O: The customer selects the product of interest. 3. Customer I/O: Signals to the customer to place the item in the Bagging Area. 4. Customer I/O: Signals to the System that an item is to be added, indicating the information about the item. 5. System: Blocks the self-checkout system from further customer interaction. 6. Bagging Area: Signals to the System that the weight has changed. 7. System: Unblocks the self-checkout system.	
Exceptions: 1. The weight in the Bagging Area does not correspond to expectations; see **Weight Discrepancy**.	
Priority:	High, must be implemented for the system to function-well.
When available:	Third iteration.

Frequency of use:	Many times per day.
Channel to actor:	Hardwired.
Secondary actors:	Bagging Area.
Channels to secondary actors: Bagging Area: Hardwired.	
Open issues: 1. The form of the visual catalogue and the manner in which one could browse it are vague. Should it be in alphabetical order, where all products starting with a given letter are displayed? Should it be done like in Netflix, arranged roughly be category with a long list of images that can be scrolled through?	

Use case:	*Add Item by PLU Code*
Primary actor:	Customer I/O
Goal in context:	To permit the customer to type a known PLU code in order to add an item.
Preconditions:	The system is expecting a customer to enter a PLU code.
Trigger:	The customer has an item with a PLU code and they want to add this item to their bill.
Scenario: 1. Customer I/O: Displays a virtual numeric keyboard. 2. Customer I/O: Accepts the customer's input of the PLU code. 3. Customer I/O: Signals to the System that an item with a given PLU code is being added. 4. System: Blocks the self-checkout station from further input. 5. Customer I/O: Signals to the customer to add the item to the Bagging Area. 6. Bagging Area: Signals the System that the weight has changed. 7. System: Unblocks the station. 8. Customer I/O: Returns to its standard display.	
Exceptions: 1. The weight in the Bagging Area does not correspond to expectations; see **Weight Discrepancy**.	
Priority:	High, must be implemented.
When available:	Third iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired.
Secondary actors:	Bagging Area.
Channels to secondary actors: Bagging Area: Hardwired.	
Open issues: 1. Simple mechanisms for self-correction by the customer, like a backspace key, should be supported. 2. The customer ought to have the opportunity to see that the PLU does not correspond to their item. At this point, communication with the attendant is likely the best option for correcting the problem.	

Use case:	*Do Not Place Item in Bagging Area*
Primary actor:	Customer I/O.
Goal in context:	To permit the customer to avoid placing an item in the bagging area, presumably because it is too big or too heavy.
Preconditions:	The system is expecting the customer to place an item in the bagging area.

Trigger:	The customer signals to the Customer I/O that they do not want to bag their item.
Scenario: 1. System: Blocks the self-checkout station from further customer input. 2. System: Signals to the Attendant I/O that a no-bagging request is in progress. 3. Attendant I/O: Signals to the System that the request is approved. 4. System: Reduces the expected weight in the Bagging Area by the expected weight of the item. 5. System: Unblocks the station.	
Exceptions: 1. The customer adds the item to the Bagging Area anyways; see **Weight Discrepancy**.	
Priority:	Medium, should be implemented to avoid legal consequences of a customer becoming injured due to moving a heavy item or as a result of collision with a bulky item.
When available:	Third iteration.
Frequency of use:	A few times per week.
Channel to actor:	Hardwired.
Secondary actors:	Attendant I/O.
Channels to secondary actors: Attendant I/O: Local area network.	
Open issues: 1. Care has to be taken to ensure that "blocking the station" does not inhibit the customer from making this request.	

Use case:	*Weight Discrepancy*
Primary actor:	None.
Goal in context:	To react to a difference in the actual weight of the bagging area and the expected weight, due to items being moved improperly into or out of the bagging area, or due to true weights differing from ideal weights or measured weights.
Preconditions:	The system is in the midst of a customer's session.
Trigger:	The Bagging Area informs the System of its current weight, and the System deems that this weight is unacceptably different from expectations.
Scenario: 1. System: Blocks the self-checkout station from further customer interaction. 2. System: Signals the Customer I/O regarding the weight discrepancy. 3. System: Signals the Attendant I/O regarding the weight discrepancy. There are three options: a. Bagging Area: Notifies the System about a weight change [if the customer adds or removes an item in response]; OR, b. Customer I/O: Signals the System about a do-not-bag request (see **Do Not Place Item in Bagging Area**); OR, c. Attendant I/O: Signals the System of a weight-discrepancy approval.	
Exceptions: 1. If the attendant does not approve the weight discrepancy, they will need to manually investigate what has been placed in the Bagging Area and what is supposed to be there. 2. If the weight change from the Bagging Area still does not concur with expectations, the weight discrepancy will continue.	
Priority:	High, must be implemented for the system to function.

When available:	Second iteration.
Frequency of use:	Potentially constant.
Channel to actor:	N/A.
Secondary actors:	Bagging Area, Customer I/O, Attendant I/O.
Channels to secondary actors: Bagging Area: Hardwired. Customer I/O: Hardwired. Attendant I/O: Local area network.	
Open issues: 1. If the scale is untrustworthy, spurious weight discrepancies will occur, upsetting customers and attendants alike. 2. Scales cannot be trusted 100%, so it is important that attendants be attentive to the stations, and that they be provided with the means to correct problems.	

Use case:	*Select Language*
Primary actor:	Customer I/O.
Goal in context:	To allow the customer to interact with the system in a language of their choice.
Preconditions:	The system is ready for customer input.
Trigger:	The customer wishes to interact with the system in a particular language.
Scenario: 1. Customer I/O: Displays the language options available at the station. 2. Customer I/O: Signals to the System the selected language. 3. Customer I/O: Ready for further customer interaction.	
Exceptions: 1. If a desired language is not present, or if the customer changes their mind, the option to cancel should be available.	
Priority:	Low, may be skipped without a major impact on system usability.
When available:	Third iteration.
Frequency of use:	Infrequent; a few times per week.
Channel to actor:	Hardwired.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: 1. How many languages must be supported? 2. Is the standard language to be English unless something else is selected? Or should this be a configuration setting when the system is installed? 3. How will this interact with the customer if they are vision-impaired? 4. The cost of maintaining the system and its attendant databases in additional languages could lead to unacceptably high costs. Poor translation could lead to customer misunderstanding or offense.	

Use case:	*Select Audio I/O*
Primary actor:	Customer I/O.
Goal in context:	To allow the customer to interact with the system in an auditory manner, speaking into a microphone and hearing from a speaker.

Preconditions:	The system is ready for customer input.
Trigger:	The customer desires to interact with the system verbally.
Scenario: 1. Customer I/O: Signals to the System that verbal interaction is to be used.	
Exceptions: None.	
Priority:	Low, as most customers can be expected to prefer verbal interaction. However, failure to provide this functionality could be legally challenged as discriminatory.
When available:	Third iteration.
Frequency of use:	Once a week.
Channel to actor:	Hardwired.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: 1. Interacting with all the functions of the system verbally could be challenging, requiring significantly different user interface design. 2. Multiple stations using verbal interaction could lead to confusion for the system and for customers, especially if volume levels need to be elevated for some customers. 3. The costs of this additional development could be significant compared to the costs of the base functionality. 4. Presumably the customer should be able to change their mind and also switch back to visual/touch based interaction. 5. Should attendants also be able to interact verbally with their station? Failure to support this could be deemed as discriminatory hiring practices.	

Use case:	*Prevent Station Use*
Primary actor:	Attendant I/O.
Goal in context:	To prevent a station from being used by customers while it is undergoing maintenance.
Preconditions:	The system is otherwise ready for customer interaction. The station to suspend is not in the midst of a customer session.
Trigger:	The attendant needs to perform some sort of maintenance on the station.
Scenario: 1. Attendant I/O: The stations that can be suspended are indicated. 2. Attendant I/O: The specific station to suspend is selected. 3. Attendant I/O: Signals to the System to suspend the indicated station. 4. System: Signals to the Customer I/O of the indicated station that its interaction is suspended, preventing customer interaction.	
Exceptions: 1. The attendant should not be able to suspend a station in the midst of a customer session. If there is a technical problem with the station in the middle of a session, this may be unavoidable, and so the attendant should be able to force suspension even during a session. This should not end the session, so that the problem could be corrected, the station un-suspended, and session resumed without alteration.	
Priority:	Medium, necessary for maintenance purposes.

When available:	Third iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired.
Secondary actors:	Customer I/O.
Channels to secondary actors: Customer I/O: Local area network.	
Open issues: 1. In some situations, it might make sense to simply cancel the customer session. There isn't currently a use case to support that.	

Use case:	*Permit Station Use*
Primary actor:	Attendant I/O.
Goal in context:	To again permit the use of a station by customers, after it has undergone some sort of maintenance.
Preconditions:	The station to be un-suspended must be currently suspended.
Trigger:	The attendant has finished basic maintenance on a station and wants to permit it to be used by customers again.
Scenario: 1. Attendant I/O: The list of stations that can be un-suspended are displayed for selection. 2. Attendant I/O: Signals to the System which station is to be un-suspended. 3. System: Signals to the Customer I/O for the indicated station to un-suspend itself. 4. Customer I/O: Ready for further customer interaction.	
Exceptions: 1. If there is some problem by which the system cannot communicate with a station, it will have to remain suspended. By this can be enforced through physical protocols, like placing an "Out of Order" sign on the station.	
Priority:	Medium, necessary for maintenance purposes.
When available:	Third iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired.
Secondary actors:	Customer I/O.
Channels to secondary actors: Customer I/O: Local area network.	
Open issues: None.	

Use case:	*Pay*
Primary actor:	Customer I/O.
Goal in context:	To permit the customer to pay for their items.
Preconditions:	There are items on the customer's bill for which sufficient payment has yet to be made.
Trigger:	The customer wishes to pay for their items.
Scenario: 1. (Abstract use case) The customer indicates the mode of payment that they want to use.	

2. (Abstract use case) The customer provides payment, signalling this to the System. 3. System: The amount due is reduced by the payment. 4. System: Signals to the Customer I/O the remaining amount due. 5. Customer I/O: Update the amount due. 6. Customer I/O: Ready for additional customer interaction.	
Exceptions: 1. If the payment is not made or not accepted, the remaining amount due is not reduced, but the customer session is otherwise unaffected.	
Priority:	High, must be implemented.
When available:	First iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Hardwired.
Secondary actors:	Unknown.
Channels to secondary actors: N/A.	
Open issues: 1. Should the customer be forced to make full payment, or can they make partial payment and then return to adding/removing items?	

Use case:	*Pay with Cash*
Primary actor:	Cash I/O.
Goal in context:	To permit the customer to provide coins and/or banknotes as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay cash for their bill.
Scenario: 1. Cash I/O: Signals the insertion of coins and banknotes to the System. 2. System: Reduces the remaining amount due by the value of the inserted cash. 3. System: Signals to the Customer I/O the updated amount due after the insertion of each coin or banknote. 4. Customer I/O: Updates the amount due displayed to the customer. 5. System: If the remaining amount due is greater than 0, go to 1. 6. System: If the remaining amount due is less than 0, signal to Cash I/O the amount of change due. 7. Cash I/O: Dispense the change due to the customer. 8. Once payment in full is made and change returned to the customer, see **Print Receipt**.	
Exceptions: 1. If the customer inserts cash that is deemed unacceptable, this will be returned to the customer without involving the System, presumably handled in hardware. 2. If insufficient change is available, the attendant should be signalled as to the change still due to the customer and the station should be suspended so that maintenance can be conducted on it.	
Priority:	Medium, essential in some business contexts, undesired in others.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired.
Secondary actors:	Customer I/O.
Channels to secondary actors: Customer I/O: Hardwired.	

Open issues: 1. The hardware has to handle invalid cash, to reject it without involving the control software. 2. Should mixed modes of payment be supported (e.g., part cash, part crypto)?

Use case:	*Pay with Credit*
Primary actor:	Customer I/O.
Goal in context:	To permit the customer to provide a credit card as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via credit card for their bill.
Scenario: 1. Customer I/O: Signals the insertion of a credit card and PIN. 2. System: Validates the PIN against the credit card. 3. System: Signals to the Bank the details of the credit card and the amount to be charged. 4. Bank: Signals to the System the hold number against the account of the credit card. 5. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of credit available. 6. Bank: Signals to the System that the transaction was successful. 7. Customer I/O: Updates the amount due displayed to the customer. 8. <**Print Receipt extension point**>.	
Exceptions: 1. If the customer enters the wrong PIN three times in a row, the card will be blocked. 2. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	First iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: None.	

Use case:	*Pay with Crypto*
Primary actor:	Crypto Authority
Goal in context:	To allow the customer to pay their bill with cryptocurrency.
Preconditions:	The system is ready to receive customer payment.
Trigger:	The customer wishes to pay with cryptocurrency.
Scenario: 1. Customer I/O: Signals to System that a cryptocurrency payment is to be made. 2. System: Communicates with Crypto Authority regarding the payment. 3. Crypto Authority: Authorizes the payment? 4. System: Reduces the remaining amount due by the amount authorized by the Crypto Authority? 5. System: Signals to the Customer I/O that the remaining amount due has been reduced. 6. Customer I/O: Updates the remaining amount due.	
Exceptions:	

1. If communication with the Crypto Authority is not established securely, the transaction should be cancelled and the remaining amount due will not be changed.	
Priority:	Low, may be skipped since the number of customers wishing to use this is likely to be few.
When available:	Third iteration.
Frequency of use:	Infrequent, a few times per week.
Channel to actor:	Internet.
Secondary actors:	Customer I/O.
Channels to secondary actors: Customer I/O: Hardwired.	
Open issues: 1. What kinds of cryptocurrency are to be supported? 2. What kind of hardware is needed to permit the customer to provide details of their cryptocurrency? 3. How is conversion between crypto and local currency to be supported? 4. What protocols are to be followed? 5. How is security to be handled?	

Use case:	*Print Receipt*
Primary actor:	None.
Goal in context:	To provide a receipt for the purchases and payments made by the customer.
Preconditions:	Payment in full has been received for the customer's bill.
Trigger:	Payment in full has been received for the customer's bill.
Scenario: 1. System: The bill record will be updated with details of the payment(s). 2. System: Signals to the Receipt Printer to print the bill record. 3. Receipt Printer: Prints the receipt. 4. System: Signals to Customer I/O that the customer's session is complete. 5. Customer I/O: Thanks the Customer. 6. Customer I/O: Ready for a new customer session.	
Exceptions: 1. If the Receipt Printer runs out of paper or ink in the middle of printing the receipt, the printing will be aborted, the station will be suspended, and the attendant informed that a duplicate receipt must be printed and that the station needs maintenance.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	N/A.
Secondary actors:	Receipt Printer, Customer I/O, Attendant I/O.
Channels to secondary actors: Receipt Printer: Hardwired. Customer I/O: Hardwired. Attendant I/O: Local area network.	
Open issues: 1. Should electronic receipts also be supported? 2. If a problem occurs after payment is made, how can a duplicate receipt be created for the customer? 3. Can the customer decline printing a receipt?	