

Specification

DRUPAL E-COMMERCE
Recurring Payments, Role Assignments and User Account Provisions

Version 4.7

Copyright © 2006 Synerger Pty Ltd



Synerger

Document Version	Revision Date	Changes Made
4.7.1	4 Jul 2006	- Initially published
4.7.2	23 Aug 2006	<ul style="list-style-type: none"> - Changed Subscriptions to Recurring Payments - Changed Roles to Role Assignments - Changed User Accounts to User Account Provisions - Added more requirements gathered from discussions to date - Converted a checkbox in Role Assignments to be roles assigned at expiry - Added Appendix A: discussion of why implementation of concrete recurring payment code is undesired - Decided on one of the User Account Provision implementations and moved other suggestions to Appendix B
4.7.3	4 Sep 2006	<ul style="list-style-type: none"> - Replace 'on first purchase' hook with an 'on purchase' hook with a boolean flag parameter set to TRUE when it is a first purchase - Redesigned reminder mails - Redesigned expiry schedules - Acknowledged importance of pro-rata support - Added spam avoidance mechanism to user account provisions

Contents

Introduction.....2

Current State of E-Commerce3

 Recurring Payments 3

 Role Assignments 3

 User Account Provisions..... 3

Proposed Changes.....4

 Recurring Payments 4

 Role Assignments 4

 User Account Provisions..... 5

Summary of Proposed Changes.....6

 Recurring Payments 6

 Role Assignments 6

 User Account Provisions..... 6

Other Requirements7

What Can I Do To Help?.....9

Appendix A.....10

Appendix B.....11

Introduction

This is a specification for proposed changes to Drupal E-Commerce in the realms of *recurring payments*, *role alterations* and *user account provision*. To give readers some idea about what these terms mean some definitions are in order.

Some sites sell time-limited memberships giving rise to repetitive transactions for each membership period. In the context of this document, **recurring payments** are a way to automate these transactions.

Roles are Drupal's way of grouping access rights and tying these in with the E-Commerce system means customers can purchase rights to access site features. A customer will then be purchasing roles and at the time of transaction **role assignments** occur to assign the new role(s) to the customer.

User accounts are the rights of a user to login to a site coupled with their user profile. Sites may wish to only provide site access to those who have paid for it. **User account provisions** occur when payments for site access have been received and a user account is created then provided to the customer.

This specification explains the current state of Drupal E-Commerce as of version 4.7. Reasons why the changes are proposed are given where appropriate. It describes the proposed changes followed by a summary of those changes, which can be used as a quick reference. Other requirements have been stated by stakeholders and they are listed in the penultimate section. Finally, some details are given about how the reader can assist with the changes.

An appendix at the end of this document explains our reasons for not implementing recurring payment support into payment gateway modules.

It is important to note that this document attempts to avoid discussion of implementation specifics unless duly warranted such as when a certain direction of implementation is advised. Equally important is that implementation of the functionality herein will take efficiency into consideration but a guarantee the most efficient solution is not required. At this stage, it is more important to have the functionality than to spend time on honing the solution.

Stakeholders in the E-Commerce system are asked to read through the document and provide comments.

Current State of E-Commerce

Drupal E-Commerce currently has some of the functionality for two of the three components. We now examine the current state of the E-Commerce system and why the changes need to be made. We look specifically at *recurring payments*, *role assignments* and *user account provisions*.

Recurring Payments

E-Commerce currently has recurring product support built-in where products can be set to have a product-specific lifetime. These features are part of the E-Commerce core or, in other words, closely tied with the core E-Commerce code.

Renewal email schedules are set system-wide rather than per-product making it impossible to have products with different reminder schedules. For example, a reminder for product A sent one month before expiry and the reminder for product B sent 2 days before expiry.

Renewal email content is also set system-wide so per-product reminders are not possible.

Recurring payments are currently manual transactions where the customer must enter in payment information. Support for recurring payments would make it possible to automatically charge a customer's credit card and renew their subscriptions automatically. Two methods exist for recurring payments: Paypal Subscriptions and credit card payments through some payment gateways. At this stage we are not making any commitments to include support for recurring credit card payments in the payment modules. Instead we're planning to build in the required API to support recurring payments so anyone wishing to implement it can do so. See Appendix A for information about why we aren't building the payment modules.

As of version 4.7, expiry dates are calculated from the payment date rather than the transaction date to allow room for delays in payments such as COD or cheques. We've been informed of a bug in E-Commerce where renewal expiry dates paid before the previous expiry calculate the new expiry date incorrectly effectively losing the number of days until the next expiry.

Role Assignments

User roles can be modified at checkout time with E-Commerce 4.7 with one limitation - the roles to be modified are set on a system-wide basis. This restriction causes the system to only allow one subscription level out-of-the-box. Sites requiring multiple subscription levels have to implement their own code for it.

Since 4.7, role changes are effected when payment is received rather than when the transaction is saved in a similar fashion to *recurring payments*.

User Account Provisions

There is currently no built-in support for creating user accounts when selected (by the store admin) products are purchased. A workaround does exist. Set registrations to be available without moderation, select a role (other than Authenticated User) for change at purchase time and set the product to only be available to logged in users. This is, of course, a long way to achieve the right goal and is only possible when site owners don't want to limit site registrations to only those that have purchased the products.

There is a strong push towards having user account creation available outside E-Commerce and not included in E-Commerce at all. An example workflow would be to send a user to logintobogon to create an account during checkout. Of course, this means the account will be created prior to payment being received and some businesses prefer an alternative workflow, even if the authenticated user role has no rights whatsoever. As this is a point of contention this has still been included in the specification subject to further review through comments by stakeholders.

Proposed Changes

While the following proposal for changes to the system is indeed thorough it is important that these remain the subject of scrutiny. Any valid amendment requests will be considered and incorporated if they fit with the majority of stakeholders.

A common theme to keep in mind while contemplating these changes is the need for E-Commerce to have a comprehensive event driven model built in to expose events such as product expiry. This will allow module developers to extend built-in features easily where they see fit. One of the long term goals is to build an event model that makes Drupal surpass its competition in not just efficiency but extendibility. For this to be successful the core of E-Commerce itself will need to expose the events for all workflow state alterations as well as payment state alterations. Modules, such as those implementing recurring payments, will also need to fire relevant events to expose their workflows.

Recurring Payments

A new module, `ec_recurring.module`, will be created. Recurring payment support will be removed from E-Commerce core and placed into a contrib module. Doing so will simplify the core code for the next stage of development. We will examine the feasibility of reintegrating the module into E-Commerce core once recurring payment support has stabilized.

E-Commerce will have three new **recurringapi** hooks: 'on purchase', 'on renewal' and 'on expiry'. This will allow custom modules to react to these events. The 'on purchase' hook will have a mandatory boolean parameter that is TRUE when the customer first purchases from the store.

Reminder emails will be extensively configurable. Email content will be stored and edited independently of delivery schedules. This will make it possible to either reuse content for multiple reminders or to have different content in every reminder. An Add/Edit Reminder Email page will be created so administrators can add and modify emails. Emails will have a subject and content stored along with a name set by the administrator for reference when adding mails to expiry schedules as discussed below. Another module, `ec_mail.module`, will be created and may end up used system-wide as of E-Commerce 5.0.

Expiry schedules will be created independently of products on an Add/Edit Expiry Schedule page. Expiry schedules will be named by the administrator and names listed in a list box on product edit pages. When editing an expiry schedule, the administrator will be able to add reminder emails at their choice of time. The time will be relative to the expiry date. If the selected reminder time is before the product was purchased the email won't be added and an error will be displayed.

Expiry dates recorded by the system will be tested extensively to ensure they are calculated correctly. Expiry dates will be stored with the product purchase record.

Support for pro-rata payments will not be added to the system at this stage. It is possible, though not guaranteed, that API will be created so this can be added by developers. We acknowledge that it is an important feature and efforts will be made to include this into the 5.0 release.

Role Assignments

A new module, `ec_roles.module`, will be created. Role assignment support will be removed from E-Commerce core and placed into a contrib. module. Doing so will simplify the core code for the next stage of development. We will examine the feasibility of reintegrating the module into E-Commerce core once role assignment support has stabilized.

Two role selection widgets (role selectors) will be added to the product edit page. The first role selector will be for selecting the roles assigned for each product when they are purchased. By default, all roles will be unchecked. If the product also has recurring payments enabled the roles selected here will be removed from the user when the product expires.

The second role selector will be for selecting roles assigned to the user when the product expires. This is in addition to the operation just mentioned.

User Account Provisions

A new module, `ec_useracc.module`, will be created. We will examine the feasibility of integrating the module into E-Commerce core once user account provision support has stabilized.

User account provision is a non-trivial addition to E-Commerce because it requires integration with the registration and user systems of Drupal where security and data integrity are paramount. For this reason Drupal core code for registration will be reused wherever possible.

A checkbox ("Purchase of this product creates a user account") will be added as a product attribute so each product can individually be marked as creational or non-creational. This is vital to having a system where site memberships are sold alongside other tangible products.

Another checkbox ("Block expired accounts") will be added to ensure the user account module will deal with product expirations correctly. The checkbox will not be visible if the *Recurring Payments* module is disabled. This checkbox will be unchecked and disabled if the product is non-creational.

After payment has been received a username and password are selected by the system. This will be of the form `customer_N` where N is the user ID of the new user account. If, by coincidence, some user already has this username, the system will simply keep adding underscore characters between `customer` and N until the selected username is available for use. An email detailing the login details will be sent to the customer's email address. The next stage of the process depends on whether the product is free or if money has been transacted.

Should the product be free, such as a trial membership, a page will be displayed to the customer instructing them to confirm their email address. Otherwise, if a payment was processed in the customer's session, the session is then converted to a logged in one and redirected to the account edit page (`user/edit`). They can then change their username to something more desirable and view their order information. In the case where the customer paid COD (e.g cash or cheque), the user is simply redirected back to the order administration page.

During checkout the user will be asked to either login or provide an email address. On provision of an email address the workflow just described will be followed. Should the user provide valid login details and the account is not blocked they will be logged in, checkout will continue and the account creation workflow above will be skipped. Should the account be blocked checkout will continue and the account unblocked instead of an account created.

It's important to note that other user login modules, such as `logintoboggan`, will be explored before implementing this module. We will try to reuse as much as possible as we move to a more actions/workflow centric model inside E-Commerce.

Two other alternative implementations were suggested and were not selected because of their complexity. They are given in Appendix B.

Summary of Proposed Changes

Following is a summary listing of the proposed changes. See the previous section for more information about each one.

Recurring Payments

- Create `ec_recurring.module` and `ec_mail.module`
- Add page to create and edit email templates where each template has a name, subject and content
- Add page to create and edit system-wide expiry schedules where each schedule has a name, schedule and zero or more reminder emails with associated schedules
- Add 'on purchase', 'on expiry', 'on renew' hooks to handle purchase and expiry events
- Remove current recurring payments code from E-Commerce core

Role Assignments

- Create `ec_roles.module`
- Add role selector to product edit page for selecting roles assigned to customers when products are purchased
- Add role selector to product edit page for selecting roles assigned to customers when products expire
- Remove current role code from E-Commerce core

User Account Provisions

- Create `ec_useracc.module`
- Add creational checkbox to product edit pages so product purchases result in account provision
- Add 'block on expiry' checkbox to product edit pages to select whether user accounts are blocked on expiry
- Upon payment reception a user account is created with unique username and random password set by the system
- Purchase resulting in account provision is associated with new user account
- System handles users that have expired accounts
- Force mandatory email address confirmation when free creational product is purchased

Other Requirements

Some other requirements were received in the issue posting¹ on drupal.org and in the E-Commerce Group² and they are worth mentioning here. Each requirement below has a comment ID, author, abridged requirement and proposed solution. If the comment ID reads **g.d.o** it refers to the E-Commerce Group otherwise the number is the comment ID in the drupal.org issue posting.

Comment ID	Author	Requirement
#11	@wpd	Buy membership but have purchase affect another user account
Solution: Introduce gift subscriptions using the coupon system otherwise this can be done once the implementation of above user account specification is completed by supplying the other user's email address at checkout and the account username/password will be sent to them.		
#14	@coupet	Recurring payments have setup costs
#14	@coupet	Require a previous purchase of a specific product before allowing purchase of another
Solution: Introduce product dependencies into the system.		
#14	@coupet	Recurrence schedule selected by user
Solution: Create sub-products for each different recurrence schedule		
#14	@coupet	Trial purchases allowing different (e.g. introductory) pricing
#14	@coupet	Different user categories have different versions of the product
g.d.o	@warco	Renewal pricing different from normal price
Solution: Introduce role-based purchase restrictions and settings		
#14	@coupet	Each product has unlimited attributes one can collect from the user
NO SOLUTION PROPOSED YET		
#15	@coupet	Generate notices for invoice, payment due, payment overdue, service suspension, payment declined, payment received
Solution: Using the specified framework to build the support required for subscriptions, the addition of notices will be very intuitive		
#14	@coupet	Free access to a product once another product is purchased
#14	@coupet	Grant extra role for specified period of time with purchase of a product
Solution: Introduce product associations (extension of dependencies) so products are linked to other products. In the cart view, these product associations can then facilitate display of products you get for free, optional (or suggested) products, recommended products or upgrades. In order for this to work the link direction must also be set so you can have one-way dependencies and co-dependencies.		

¹ <http://drupal.org/node/53888> - move subscriptions to own module

² <http://groups.drupal.org/node/952> - Specification for changes to E-commerce subscriptions (recurring memberships), roles and user accounts

Comment ID	Author	Requirement
g.d.o	@warco	Administration reports for subscriptions
Solution: This will be possible to implement after the functionality, detailed in this specification, is added.		

What Can I Do To Help?

This specification is intended to give the community an idea of what is proposed for Drupal E-Commerce and to also specify what is required for inclusion of contributed code should someone want their subscription/role/user account code added to Drupal E-Commerce.

Your contribution is very important to the successful progress of Drupal E-Commerce. For progress with this specification please provide either supporting comments or clearly written addendums to this specification so we can encompass as many different configurations as possible.

Appendix A

We now look at why recurring payment implementation for specific gateways will be omitted from Drupal.

There are many security risks involved when implementing recurring payment solutions. First and foremost is the storage of credit card details in a web accessible system. If that system becomes compromised the store owner is liable and with some crafty legal support (in some countries) they may even be able to pin the liability on the developers.

Some gateways implement mechanisms not requiring storage of credit card details and we're yet to find a gateway that will ensure people administering and developing Drupal solutions aren't exposed to undue financial risk without their knowledge. We haven't exhaustively looked into this problem at this stage and we plan to once the recurring payment API is built in.

Appendix B

Two other implementations of the user account provisions workflow were suggested. They are shown below.

1. The system will request a username during the checkout process. This will happen before the review page and only when the user is not logged in. At this point the username entered will be compared using the rules used in registration to ensure it's not a duplicate. If the email address and the username entered are the same as one already recorded in the system a login screen is displayed with the username already filled in. After login or entry of valid new username the checkout procedure continues to the review page. The review page will show the user the username they've selected or are using. Payment process then begins.

In the above process the time between when the username is collected and the payment is received (thus user account created) can be any length. To avoid the certain race condition it is important to have some way to prevent other users selecting the same username during this time. Ideally this would also prevent username duplication in registrations unrelated to E-Commerce. One such way would be to create a blocked account for the user with some attached data linking it to the transaction taking place.

Upon successful payment a user account is created/unblocked, the transaction is linked to that user, a registration/confirmation email is sent to them, logs entry is added to show a new account was created and they are automatically logged in going to the user profile editing page allowing them to adjust their password. Of course, the automatic login is only performed when the user is not logged in already.

2. One other possible scenario is not collecting a username during checkout and instead the user follows a link in the purchase confirmation email to a special registration page exposed by E-Commerce. This will allow email address confirmation prior to the user gaining access to the system giving an additional security precaution. This will require some special handling for expired accounts being renewed (keep in mind that expired accounts may have blocked accounts).

This method will require some extra handling for previous account holders wishing to reuse their account after renewing.