

## Summary: DrupalCon SF – April 19-21/10

- **Attendees:** Eva Grabinski (Waterloo CMS Project Lead; IST), Pat Lafranier (IST), Terry Stewart (AHS), and Guillermo Fuentes (ARTS)
  - World bi-annual conference with **3000 attendees** at the Moscone Center, San Francisco.
  - **3 keynotes:**
    - Dries Buytaert, founder of Drupal, "The State of Drupal"
    - Tim O'Reilly, founder of O'Reilly Media, "Open Source in the Cloud Era"
    - David Cole, whitehouse.gov, "Open Source in Government"
  - **7 tracks:** welcome to Drupal; design, theme and usability; under the hood, configuration, setup and administration; leveraging Drupal for your business; providing professional Drupal Services.
  - Day **workshops** – April 18; attended by Eva and Terry
  - **Birds of a Feather** (BOFs)
  - **Sponsor Showcase** – 50+ companies (Lullabot, Acquia, Google, Commerce Guys, Microsoft ...)
  - Evening parties , coder lounges – not attended
  - **Website:** [sf2010.drupal.org](http://sf2010.drupal.org)
- 

### Terry's notes <stewart@uwaterloo.ca>

#### DRUPAL SECURITY SEMINAR, Sunday, April 18, 2010, Greg Knaddison

- Most of presentation in the book. This summary touches only on highlights and comments not in the book. The book is better than I could ever describe here. This was also a hands-on seminar; so much time was spent on examining code.
- Greg Knaddison is author of "Cracking Drupal- a Drop in the Bucket". Also <http://crackingdrupal.com/>
- There is a Drupal Security Module at: [http://drupal.org/project/security\\_review](http://drupal.org/project/security_review)
  - It is a first attempt and needs work but still useful
- There is a Drupal Password Policy module at: [http://drupal.org/project/password\\_policy](http://drupal.org/project/password_policy)
- There is a Drupal Security Team
  - Main page: <http://drupal.org/security>
  - Volunteers; experts in various aspects; core members part of security team;
  - Issue security advisories;
  - Maintain mailing list and RSS feeds;
  - Process: when a vulnerability is reported to Security team, it is not reported until a fix is available;
  - Education component
  - Documentation tasks
- Only one worm that ever affected Drupal and an update was available 4 months before attack, so it only affect unpatched systems
- Discussed owasp.org
  - Open-source web application security program (OWASP)
  - Track top 10 worries
  - Top 10 for Drupal include: Code injection, XSS, Weak authentication, direct object reference, unvalidated redirects, insecure cryptographic storage and insecure transport layer (SSL – outside of Drupal control)
- Drupal 6 relies on MD5 whereas Drupal 7 is a pluggable framework and the salt is separate from the database;
- "truly destructive actions should be hard and should never really truly delete (ie, backups taken before);
- Big four are still a problem and issues generally outside the control of Drupal
- Access By-pass – missing or inadequate security checks (ie, requester allowed to perform action without authentication; not using user\_access('<permission>') on calls
- CSRF (Cross-site Scripting Forgeries) – taking action without confirming consent. One common place is in Ajax table requests; also, beware of URL shorteners and img tags. POST v. GET – post for changing data; get for retrieving! Use FAPI (only slight overhead)

- XSS (Cross-site Scripting). Test out by putting js in forms `<script>(alert 'XSS problem');</script>`; filtered HTML should be the default; html corrector should be last; XSS for themers/coders/admin. Good video at: <http://highvisibilitywebsites.com/drupal-security-video-example-user-account-hijacking-xss>
- Injection. Input data contains data to trick system into shifting context. Eg, comment has js; Eg, 'select \* from table where title=\$title', many other examples.
- Secure pages module: <http://drupal.org/project/securepages>
- Hijack prevention module: [http://drupal.org/project/securepages\\_prevent\\_hijack](http://drupal.org/project/securepages_prevent_hijack)
- Could use SSL for all pages – 15% hit on server and client side.
- Secure modules: the more popular, the more likely safe.
- Secure module writing <http://drupal.org/writing-secure-code>
- Drush has core update but not perfected – blows away versioning
- Drupal 7 has concept of public/private files/images
- Securing user input, see: <http://drupal.org/security/secure-configuration>

## **ACCESSIBILITY IN DRUPAL 6 & 7, Monday, April 19, 2010, Katherine Lynch (Drexler)**

<http://sf2010.drupal.org/conference/sessions/accessibility-drupal-6-and-drupal-7-write-accessible-modules-and-themes>

*Great session covering fundamental of Accessibility*

- Fundamentals
  - “web accessibility means people can use the web”
  - W3C/WAI/WCA/ARIA/AT,ATs
  - a11y is short form used in twitter
  - considerations: vision, hearing, speech, motor, cognition, hearing, aging
  - some stats: “more disabled users than there are users of any 1 browser!”
  - types of assistive technologies: screen reader, no mouse, Braille, screen magnification, special keyboards, hearing assistance, cognitive problems, mouth input sticks, head mounts, blow tube, etc. etc. etc.
  - a11y is for everyone – cleaner code, smarter, semantic, valid, machine readable
  - How:
    - WCAG2.0 documents – html, css, screenreaders, etc.
    - Sites should conform to POUR (P=perceivable to all senses; O=operable interface (flexible); U=understandable (standard actions/reading comprehension level); R=robust (as technology advances))
- Accessible Themes
  - Start with a11y in mind; squint test; Zen theme good start for a11y;
  - Color – contrast ratio 3:1; color only cues bad; use std link colours; WCAG sets out colors;
  - Alt text – on all images (also good for mobile); decorative images should have alt=”; image maps OK but must use alt tags on all regions
  - Skip links – skip to regions on page; add block to theme so content authors can use include statements
  - Styling – use H1-H6; good for more than a11y; provides proper hierarchy
  - Forms – use labels, filesets, legends
  - Tables – scope ids/headers/summaries
  - Defaults in Drupal, esp. Zen, pretty good
  - Hidden content – in Drupal 7 – elements invisible v. hidden; use show/hide function;
  - Layout/Source Order – don’t assume reader will backup even if told; consider regions and source order
- Coding Modules
  - Encourage best practices in coding
  - Create a11y admin interfaces
  - Don’t have content and only 1 way out
  - Keyboard traps – no actions on click or focus only
  - Focus indicators – reset.css is bad; fixed in Drupal 7
  - Mono prompts bad
- Rich Content
  - Audio – transcripts available; don’t play audio automatically;
  - Video – text/audio equivalent
  - Use youtube
  - Drupal 7 media modules have lots of features
- ARIA
  - Not completely supported; still good to markup
  - Landmark roles – app, banner, etc
  - Determine live regions
  - No ARIA in Drupal 6
- Testing for Accessibility
  - Use screen reader on site (eg, Jaws, Nvda, Thunder)

- Users, users, users
  - Graceful degradation (gradually remove everything)
- Tools
  - Visicheck, Fangs, Firefox a11y extension, Validators
- Drupal 7 should make a11y entirely possible.

### **THE STATE OF DRUPAL AS A WEBAPP AND PRODUCT, Monday, April 19, 2010, Panel**

<http://sf2010.drupal.org/conference/sessions/state-drupal-web-application-product-platform>

*This session was about building out full products from Drupal and is not particularly relevant to UW.*

- What is a product? Lifecycle, customers, revenue, consulting, training, commitment to the future.
- Why is Drupal a good product platform?
  - Success as product not a solution
  - Being adopted as a platform for other vertical markets
  - Both framework and CMS – opportunity for verticals
  - Build quickly to show client
- Why service shops turn to products?
  - Repeatable patterns
  - Pain-points
  - Productize as you build
  - Needs to product out of the box;
  - Tired/bored doing the same thing
- Is Drupal a product or a brick in a product?
  - Eg, Drupal Gardens both
- Consider other parts of the product
  - Are all components open source?
  - How much to give back?
  - Who decides how/when to give back?
  - Acquia Theme building not open source... yet.
- Products as Platform Drivers
  - Does product influence core; modules, “small” core
- Keep Drupal small or add features?

### **ENTERPRISE TRENDS: ANALYST PANEL, Monday, April 19, 2010, Panel**

<http://sf2010.drupal.org/conference/sessions/enterprise-trends-analyst-panel-cms-experts>

*This session was interesting but was more about convincing the enterprise that Drupal is safe and good, etc.*

- DofD says open source is safe
- Public sector opening floodgates
- Large deployments
- Web as platform
- Social platforms
- Less about firewalls
- One platform for all
- Replacing old CMS platforms
- Multi-channel delivery of content (print, movie, iPod, kindle, etc)
- Legitimize channel IT; shadow IT
- Enterprise 2.0 – initiatives behind firewall that drive performance and improve employee collaboration
- Open Source v. Proprietary
  - Drupal is novel framework, more nimble; old CMS’s bloated, cumbersome;
  - Faster pace of innovation (eg, Drupal Commons, Open Atrium)
  - Not about price tag but how quickly can we change
  - Mitigate market risks with Drupal
- Commercial v. Community-based O/S
  - Eg, Acquia Drupal does not add anything you can’t download but packaged
  - Need for certification, training of community;
  - Difficult for enterprises to move from .net to LAMP
  - Best of breed v. good enough
  - Platform products need solutions
  - No product but hundreds of solutions
  - “If you focus on tools, you are one.”
  - No one has an E2 problem, they have a business problem
- Impression of Drupal

- Biggest threats to Drupal – not solid os; community a threat; keeping federation together
  - Should start small with pilot and convince
- Commercial firms starting to productize Drupal

### **CCK DEMYSTIFIED, Monday, April 19, 2010 Doug Vann**

<http://sf2010.drupal.org/conference/sessions/cck-demystified>

*This session was an introduction to using CCK... a hands on demo. I did not take notes. Nothing here that wasn't covered by the Linda.com training or the Lullabot videos.*

### **ADOPTING A STANDARD FOR THE ENTERPRISE: CASE STUDY OF A FLORIDA HOSPITAL, Monday, April 19, 2010 Rick Mann**

<http://sf2010.drupal.org/conference/sessions/adopting-standard-enterprise-social-publishing-case-study-florida-hospital>

*This session was a case study of a faith based Florida hospital with 8 campuses, 19000 employees and 1,000, 000 patient visits that deployed Drupal.*

- In 2001, intranet was just a bunch of links; no content owner; had 12000 visits a day; 80 dept sites and proprietary CMS;
- No consistency, training, site not maintained; need complete replacement;
- Small IT shop with 3 people; history of self-reliance; handful of static FTP sites;
- Goals: replace old CMS; 80/20 rule (80% with 20% effort), ease transition, increase functionality, consistent themes, easy to admin
- Considered: new version of old CMS, Liferay, Sharepoint, Drupal
- Chose Drupal because: shared skill set, shared infrastructure, implement quickly/cheaply, community, features, Acquia, easy admin
- Drupal GAP Analysis
  - Used Acquia
  - Concurrent testing and learning
  - Learned what to ask
  - How to get it approved by management: Who else is using; How does it compare; Change management plan
- How to get it approved by users:
  - Oooo.... shiny; Chance to cleanup
- Gave users: basic CMS functions, categorize/nested users, clean simple theme, banner and block freedom
- Used 11 basic modules (node access, ckeditor, ldap, role delegation, among them)
- Back-end admin tools:
  - Create a new site from template and configures modules
  - Done by copying an existing site and db as template
  - Makes Apache config changes
  - Site comes complete
- User training:
  - 2 roles – creator/site owner
  - Required training for all (140 people)
  - Timeline forced users to learn
  - Setup early adopters to pilot system and (go-to people)
  - Other training: videos, forums, blogs, feedback, lunch and learn, Q&A lab sessions
  - Intangibles to Success: MIS open to ideas, good in house talent
- Side benefits to Drupal: now have FHTV, clinician forms, med use reference, radiology team site
- Challenges: no central directions/owner, better policies on content, graphic talent on teams, demanding end users, fractured content,
- Features need to add: image management, subscriptions, forms
- Next step to move Insite (.net) to Drupal, better menus, better search (SOLR), tagged news
- Addressing sceptics: make the business case; it is cheaper, better supported, mature, staff-able, extendable, scalable, Acquia modules good, SEO

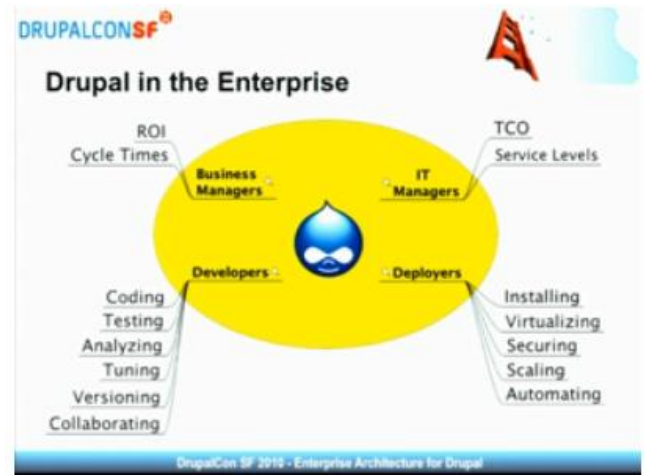
### **ENTERPRISE ARCHITECTURE FOR DRUPAL, Tuesday, April 20, 2010**

<http://sf2010.drupal.org/conference/sessions/enterprise-architecture-drupal>

*This session was very academic – more about project management than it was about system architecture – more up Eva's area.*

- "Drupal has reached the point that enterprise cannot ignore it."
- "The next frontier is Enterprise." Dries Buytaert.
- What is Enterprise Architecture (EA)? Large organization, 1000's of employees, complex IT organization.

- Architecture – description of a system and process for designing it
- “A Pattern Language.” Chris Alexander is basis for this talk
- Can’t build a city without a planner
- Gave examples of failure due to lack of EA
- Talked about two frameworks for EA
  1. Zachman Framework
  2. TOGAF Framework (he used a “Tailored” TOGAF Framework)
- What is a Requirement?
  1. A function the system must to
  2. A property a system must have
  3. A constraint a system must have
  4. Specific
  5. Measureable
  6. Attainable
  7. Realizable
  8. Traceable
- Used cube to illustrate a case study as an iterative process, vets scope; know when you’re done
- Tailored Framework
  1. Architecture vision – strategy, goals, operating model, business capacities, etc
  2. Current State – find out what’s wrong, validate with stakeholder
  3. Future State – select reference, id patterns, id risks, id gaps, build future state, validate
  4. Strategic Roadmap – prioritize gaps, create roadmap, implement plan, validate
  5. Arch Governance – id controls, effectiveness of controls, recommend change, validate
  6. Business Case – id cost/risks of current state, benchmark metrics, est. cost of future state, create cost/benefit analysis
  7. GO TO 1 and repeat



**DRUPAL GARDENS AND ACCELERATING DRUPAL ADOPTION, Tuesday, April 20, 2010 Dries Buytaert**

<http://sf2010.drupal.org/conference/sessions/drupal-gardens-design-online-15-minutes>

*This session was a demo of Drupal Gardens – I have seen the future of Web Management! This is exactly how we need to be viewing our project. We are providing a hosted service to campus.*

- Started with customer testimonials; 450 customers to date, 150 partners
- Mission to accelerate Drupal Growth
- Problems gardens addresses: too hard to start, install, theme, not enterprise, doesn’t do large sites, talent
- Demo of Gardens – Drupal 7 running as a hosted service
- Value added features:
  - Admin bar, wysiwyg editor, media, theme builder, every theme has 15 regions, links to typekit for fonts, layered CSS being used, export button to any Drupal 7 site, save as many themes as you like, help bar for videos, apply custom domain. WOW!
  - Free ... may be pay later (I have a bunch of free trial cards)
  - Repeatable information architecture
  - SSO across gardens
  - Consistent designs themes
  - Multisite management
  - Theme libraries
  - Turnkey site deployment
  - Enterprise security and support
- Coming in future
  - More modules, views/panels, theme marketplace, partner program, open API’s
  - One stop for Drupal solutions
- Overcoming Barriers (what Acquia can do)
  - Getting Started Hard – Software Publisher Programme, Open Publish, Open Scholar. Check out OpenPublish and The Scholar’s Websites Project (Harvard). Also check out Drupal Commons – a white label community platform for civic projects (free distribution)
  - Hosting Large Sites is Hard- Acquia Hosting. Rackspace partnership.
  - Finding Drupal Talent – Drupal Training programme partnered with Chapter 3; onsite training for developers, themers, users, etc.
- Not Enterprise Ready
  - 150 enterprise customers
  - Experimenting with Drupal on a .net stack
  - Acquia Drupal on Web Platform installer
  - Drupal 7 PDO drivers

- Commerce group building MS/SQL
- Project Life Cycle (Acquia provides services at each level)
  - Define
  - Select
  - Plan
  - Learn
  - Build
  - Deploy
  - Operate
- From question session: "Will Theme Builder be a Module?" Answer: Maybe. Drupal 7 not even released yet. Need more time to get it ready?

### **FACEBOOK APPS POWERED BY DRUPAL, Tuesday, April 20, 2010 Dave Cohen**

<http://sf2010.drupal.org/conference/sessions/facebook-applications-powered-drupal>

*This session was a discussion of building Drupal apps for Facebook. Interesting and may have some applicability later.*

- 2 basic app dev platforms – canvass pages; connect. Can be desktop or mobile (not Drupal dependent)
- Why care? Fb has 400,000,000 users
- **Canvass Pages** – FB Frames with canvass area that containing anything; canvass area hosted on your server; uses FBML (Facebook Markup Language)
- How to build with Drupal
  - Challenges; FBML, links to Fb; sessions by FB – no cookies!
  - FBML much like XML
- The better idea:
  - FBML solution: produce FBML at theme layer; URL's map one-to-one
  - Links to Facebook.com – URL rewriting
  - Configure fb\_settings.php (not normal cookies)
- Do – develop canvass pages like webpages; enable HTML and FBML themes; administer app on your server/admin account; customize your FBML theme
- Don't – reinvent wheel; code logic into theme; assume javascript will work (jquery doesn't work on FB pages)
- **Connect Pages** – bring FB social features to your website
  - Immediate logic and registration
  - Uses XFBML
- Do – build Drupal way; Add xmlns:fb; use XFBML in themes, blocks; call FB connect Js classes;
- Don't – user management is tricky
- **Drupal for Facebook Project** <http://drupal.org/project/fb>
- Steps:
  - Enable modules (many options)
  - Edit settings.php to accommodate session mgt (include fb\_settings.php in settings.php)
  - Create app on canvass page; get keys from Fb
  - Paste keys in config in Drupal
- Drupal for FB allows you to easily get started; removes headaches; allows you to build social features.
- Caveats
  - User management
  - In Drupal can be anonymous or registered
  - In Facebook can be registered for app or not registered for app
  - Registered for both is easy
  - Registered for app and anon in Drupal tricky – no roles, perms, and caching a problem
  - Best to get more info and register them in Drupal

### **2.8 MILLION PAGE VIEWS PER DAY/60 MILLION PER MONTH/1 SERVER, Tuesday, April 20, 2010 Khalid Baheyeldin**

<http://sf2010.drupal.org/conference/sessions/24-million-page-views-day-60-m-month-one-server>

*An excellent session from the owner of 2Bits Software, Waterloo.*

- About Khalid: hook\_watchdog(), 37 modules contributed, specializes in performance optimization, managing large sites and custom module development.
- Site sample – an entertainment site, Alexa rank 3400, Netcraft rank top 250
- In April 2009 – 400K pg/day; 2.5MM/month; 5000 users; 8,000 nodes
- In April 2010 – 2.8 MM/day; 60 MM/month; 65,000 users; 14,000 nodes
- The Challenge -site locking up; Excessive swapping;
- Hardware – regular off the rack server; unturned; no PHP accelerator; no memcache; Apache bloat; using core caching; cache table huge; one view on voting api cause very slow query; locking at table level; fast rate of growth; 4 core server
- Steps
  - Conducted performance assessment (observe, analyze, recommend)

- Install new server from scratch
  - Only needed stuff installed
  - Custom tuned Drupal
  - Removed unnecessary module (eg, stats)
  - Remove one problematic view
  - Replaced CCK with modules
- New hardware 8 core 2xQuad, 8Gb RAM, Fast disk, 64 bit Operton
- Separate spindles fore mySQL and everything else
- Ideal would be: 1 disk for O/S, 1 for Logs, 1 for Drupal, 1 for MySQL
- Used DL360GX HP Disks
- Ubuntu 8.04 LTS 64 bit
- Apache – MPM worker (threaded); fcgid
- APC 3.0.19 (installed via pecl)
- Munin for monitoring
- Awstats for stats
- Drupal config
  - No commenting on site
  - Voting on nodes enabled
  - Book module
  - Simplicity good for scalability, security, maintenance
  - 44 modules only (average is about 120 modules – one client had 231 modules)
  - 2 custom modules -- one to replace CCK
  - 2 themes
  - Modules used: captcha, image\_captcha, comment\_subject, avatar\_selection, votingapi, fiverstar, fivestar\_comment, taxonomy\_browser, nodewords, nodewords\_basic, page\_title, profile\_csv, session\_expire, token, token\_actions, views, views\_ui, views\_export, Devel, customerror, admin\_menu, memcache\_admin, blockcache\_alter, googleanalytics, fckeditor, local\_menu, pathauto
  - Block cache enabled with block\_cache\_alter module
  - CSS aggregation enabled
  - Taxonomy browser – but must be configured properly
  - Memcache – essential for site (extended with customcache.inc – add timestamps)
  - Memcache\_bins – Showed table of which bins being hit (see presentation)
  - Watch for input format in nodes and blocks
  - Created new RAW format (no line breaks)
  - Fast path cache – little know in Drupal, your .inc implements a function called page\_cache\_fastpath(), didn't use – breaks some features
  - Syslog – since Drupal 6, uses pluggable hook\_watchdog(), reduces load on server, down side – can't see log entries from web; better solution is to push log elsewhere
  - Patches – patching is bad, keep to minimum, properly managed (VCS)
  - Fast 404s – Drupal causes scalability problems, check settings.php, #76824
  - Crawlers – found some poor ones, M/S URL Control/Lucid Media; modified settings.php to return 418 error (I'm a little teapot short and stout)
  - Apache access logs got too big (excluded static files, jpg's, png's etc)
  - mySQL to separate disk spindle:
    - datadir = /data/mysql
    - innodb-file-per-table
    - innodb\_buffer\_pool\_size=256
    - long\_query\_time = 2
  - Innodb – locking problem, converted tables in Innodb
  - Monitoring – Munin
  - Elisiya cron a good replacement for core cron
  - Queuemail module used in conjunction with Elisiya
  - The Future: varnish with http headers for cache changes; two servers – one for Drupal one for db; Drupal 7 – fields now in core (more overhead); lots of optimizations
  - Lessons: Drupal can scale, no two sites the same, complexity is an abomination, simplicity is a virtue, measure and monitor, don't fly blind, diagnose before treatment.

**AN ENTERPRISE APPROACH TO SECURING YOUR DRUPAL SITE Tuesday, April 20, 2010 Acquia and General Dynamix**

<http://sf2010.drupal.org/conference/sessions/enterprise-approach-securing-your-drupal-sites>

*A discussion of broad concepts on securing Drupal.*

- How do I trust Open Source? Will I need new controls? What resources are available?
- Myths:
  - PHP less secure than ...
  - Open Source is easier to attack
  - Doesn't have versioning and documentation

- Security resources not available
- Components of Security
  - Clear system boundaries
  - Thorough risk analysis
  - Security controls that map to risk
  - Validate your controls
  - Plan to remediate issues and mitigate
- Clear System Boundaries
  - Limits the scope of what you're investigating
  - Leverage previously secured components and infrastructure
- Risk Analysis
  - Code analysis for all custom code
  - Theming focus (look at themes closely)
  - Penetration testing
  - Vulnerability assessment
  - Gap analysis
- The Right Controls
  - Done over-engineer controls
  - Leverage existing procedures/policies
  - Focus on entire system
  - Define valid plans in parallel with controls
  - Leverage Drupal community
  - Types: preventative, detective, corrective, recovery
  - Domains: admin, logical, physical
- Validate Controls
- Firewalls
  - Configured correctly? Preventative logical control ACL's
  - Outgoing important as incoming
  - Limit access to IP nr's
  - Ssh not telnet

### **TRENDS IN DESIGN AND ARCHITECTURE OF BIG DRUPAL SITES Tuesday, April 20, 2010 Nick Lewis**

<http://sf2010.drupal.org/conference/sessions/trends-design-and-architecture-big-drupal-sites>

*A poor discussion by a poor speaker... I came up with one lesson from all of this – don't fall into "paralysis by analysis"*

- Original plan was to collect data from 25 major sites; found most the same;
- Looked a non Drupal sites – didn't help
- Drupal sites are grown not built
- Drupal is a platform for software that builds a web page
- A complex system that works invariably evolved from a working simple system – Gall's Law
- Mother Drupal == Mother Earth??
  - Modules are life forms that live and evolve or die
  - Drupal is an ecosystem (do we need to know how it works?)
- Are we Drupal Doctors?
  - Lots of bad ones out there
  - Separate symptoms from diagnosis
  - Specialists needed
- Drupal sites with bad personalities are Frankensites (???)
- Trends in progress
  - Waterfall methodology – fails Drupal at each step; blueprint it before we know about it
  - "Getting Real" is about skipping all the stuff that represents your site
  - One sentence goal for your site
  - Keep iterating
  - Design parts – never the whole (contradicts Frankensite comment above)
  - Grow don't build.

### **UBERCART FOR BEGINNERS Wednesday, April 21, 2010 Andy Lowe**

<http://sf2010.drupal.org/conference/sessions/ubercart-beginners>

*An introduction to Ubercart. We will likely be using our own PHP module to send data to Beanstream.*

- Is Ubercart right for you?
  - Basic sites: digital goods, site access, subscriptions, products with lots of attributes, complex shipping, tax requirements
- What needs improvement?
  - Internationalization, reservations, POS (there is none), dependent attributes, donations, multi-site integration (complex)

- How not to choose an e-commerce system? Search Google and pick the one with most features; need to define your requirements; need to consider: requirements, features, shipping, tax, payment methods, catalogue design, checkout requirements, "special sauce" that makes you different.
- How to choose? Evaluate, install, test, got thru with customer, consider intangibles
- Prerequisites: LAMP, Drupal requirements
- Most hosters use: CURL, PHP Memory Limit Configuration (WSOD)
- Expect to pay \$20/month for hoster with Ubercart
- Installing: install from [ubercart.org](http://ubercart.org) or UberInstaller on site
- Other modules: CCK, views, ubercart-views, securepages, google\_analytics, views\_bulk\_operations, backup/migrate, pathauto, loginTobaggin, jquery\_update, Mollom
- Themes: Acquia, Prosper, etc
- The rest of the presentation was a Demo
- SSL needs to be acquired not part of Ubercart
- Credit card config – PCI compliant, debug mode not PCI compliant
- Paypal – express checkout, payments standard (2 types) see Ubercart for PayPal documentation

### **FROM UBERCART TO DRUPAL E-Commerce Wednesday, April 21, 2010 Ryan Szrama**

<http://sf2010.drupal.org/conference/sessions/ubercart-drupal-commerce>

*A discussion of Drupal E-Commerce, essentially the next version of Ubercart for Drupal 7. Good speaker.*

- Vision to be number one ecommerce solution in world
- Drupal Commerce is free
  - Layouts, community, faceted search, mobile, flexible, user management
- Core commerce api
- Principles:
  - Separate api from ui
  - Core segregates plug-ins and packaging
  - Core features uses fields, views, entities, etc
  - Structured product entry, flexible
  - Customize checkout flows
  - Stronger lateral integration/dependence on the modules
  - Strict code enforcement and documentation standards
  - The rest was a demo – see presentation link above

### **TOP 100 DRUPAL MODULES Wednesday, April 21, 2010 Deborah Fuzetto**

<http://sf2010.drupal.org/conference/sessions/top-100-useful-contributed-modules>

*A good discussion and categorization of top Drupal Modules – no demos or discussion of how they work*

- Views
  - Views - The Views module provides a flexible method for displaying content. <http://drupal.org/project/views>
  - Calendar - display any Views date field in calendar formats <http://drupal.org/project/calendar>
  - Views Bonus – a group of modules that expands views functionality [http://drupal.org/project/views\\_bonus](http://drupal.org/project/views_bonus)
  - Views Excel Export - export Views data to Excel [http://drupal.org/project/views\\_export\\_xls](http://drupal.org/project/views_export_xls)
  - Views Bulk Operations - bulk operations to be executed on the nodes displayed by a view (delete all) [http://drupal.org/project/views\\_bulk\\_operations](http://drupal.org/project/views_bulk_operations)
  - Views Gallery – Photo gallery feature [http://drupal.org/project/views\\_gallery](http://drupal.org/project/views_gallery)
  - Views Attach - provides two additional Views display plugins, "Profile" and "Node content". These displays do not display on a page but get "attached" to either a user page or nodes of specified types. [http://drupal.org/project/views\\_attach](http://drupal.org/project/views_attach)
  - Views Rotator – rotate through view items (good for ads) [http://drupal.org/project/views\\_rotator](http://drupal.org/project/views_rotator)
  - Views Slideshow - slideshow of any content (look related modules) [http://drupal.org/project/views\\_slideshow](http://drupal.org/project/views_slideshow)
  - Views Carousel – slideshow based on the carousel jQuery plugin <http://drupal.org/project/viewscarousel>
- CCK CCK - add custom fields to nodes <http://drupal.org/project/cck>
  - Imagefield - image upload field type <http://drupal.org/project/imagefield>
  - Link – link field type <http://drupal.org/project/link>
  - Filefield - universal file upload field <http://drupal.org/project/filefield>
  - Date - date/time field type <http://drupal.org/project/date>
  - Email – email upload field type <http://drupal.org/project/email>
  - Phone – phone field type <http://drupal.org/project/phone>
  - Location – adds geographic locations fields to nodes <http://drupal.org/project/location>
  - Node reference url - node reference field type [http://drupal.org/project/nodereference\\_url](http://drupal.org/project/nodereference_url)
- Content
  - Event – content type for events <http://drupal.org/project/event>

- Event Repeat – Repeating events <http://drupal.org/project/eventrepeat>
- FAQ – content type for faqs <http://drupal.org/project/faq>
- FAQ Ask – allows users to ask question and have roles as monitors [http://drupal.org/project/faq\\_ask](http://drupal.org/project/faq_ask)
- Scheduler – published nodes on specified dates <http://drupal.org/project/scheduler>
- Signup - allows users to sign up (or register, as in register for a class) for nodes of any type <http://drupal.org/project/signup>
- Webform - adds a webform nodetype <http://drupal.org/project/webform>
- Popups Api – API for building modal ajax dialogs <http://drupal.org/project/popups>
- Add to Any Share/Save/Bookmark Button - widget to allow readers to share content with social networking services <http://drupal.org/project/addtoany>
- FCKEditor - replace textarea fields with a visual HTML editor <http://drupal.org/project/fckeditor>
- Wysiwyg - allows you to use client-side editors <http://drupal.org/project/wysiwyg>
- IMCE - image/file uploader <http://drupal.org/project/imce>
- Theming/Layout
  - Panels - create customized layouts <http://drupal.org/project/panels>
  - Quick Tabs - allows you to create blocks of tabbed content <http://drupal.org/project/quicktabs>
  - Vertical Tabs - Provides vertical tabs on the node add form [http://drupal.org/project/vertical\\_tabs](http://drupal.org/project/vertical_tabs)
  - Block Theme – templates for blocks <http://drupal.org/project/blocktheme> Block Class - add classes to any block [http://drupal.org/project/block\\_class](http://drupal.org/project/block_class)
  - Collapsibleblock - makes all Drupal blocks collapsible <http://drupal.org/project/collapsibleblock>
- Images/Media imageapi – required by other modules <http://drupal.org/project/imageapi>
  - Imagecache – set presets for image processing <http://drupal.org/project/imagecache>
  - Slideshow - slideshow transforms images attached to a post into a JavaScript enabled slideshow <http://drupal.org/project/slideshow>
  - Gallery <http://drupal.org/project/gallery>
  - SWF Tools - embed flash content on your pages <http://drupal.org/project/swftools>
  - Image FUpload - ability to upload multiple images [http://drupal.org/project/image\\_fupload](http://drupal.org/project/image_fupload)
  - Thickbox - webpage UI dialog widget <http://drupal.org/project/thickbox>
  - Lightbox - script used to overlay images <http://drupal.org/project/lightbox>
  - Comparison of Lightbox-type modules <http://drupal.org/node/266126>
- Utilities
  - Token - small bits of text that can be placed into larger documents via simple placeholders <http://drupal.org/project/token>
  - Pathauto - generates path aliases automatically <http://drupal.org/project/pathauto>
  - Path Redirect - specify a redirect from one path to another path or an external URL [http://drupal.org/project/path\\_redirect](http://drupal.org/project/path_redirect)
  - Flag – Flagging system for content <http://drupal.org/project/flag>
  - Rules - define conditionally executed actions based on occurring events (workflow) <http://drupal.org/project/rules>
  - String Overrides - provides a quick and easy way to replace any text on the site <http://drupal.org/project/stringoverrides>
  - Menu Breadcrumbs – creates breadcrumbs based on the menu of the page [http://drupal.org/project/menu\\_breadcrumb](http://drupal.org/project/menu_breadcrumb)
  - Chaos tool suite - set of APIs and tools required by other modules (Panels) allows for wizardry. <http://drupal.org/project/ctools>
  - jQuery UI - A wrapper module around the jQuery UI effects library [http://drupal.org/project/jquery\\_ui](http://drupal.org/project/jquery_ui)
  - jCarousel - allows developers and themers to make use of the jCarousel jQuery plugin <http://drupal.org/project/jcarousel>
  - Context - manage contexts for different portions of your site then do things based on what context you are in <http://drupal.org/project/context>
  - Features – collection of Drupal entities (views, contexts, CCK fields, imagecache presets) <http://drupal.org/project/features>
  - Spaces – allows features to be enabled and customized in different spaces <http://drupal.org/project/spaces>
- Development
  - Devel – a bunch of helpful modules useful when building the site <http://drupal.org/project/devel>
  - Drush – command line shell and Unix scripting interface for Drupal <http://drupal.org/project/drush>
  - Coder - assists with code review and version upgrade <http://drupal.org/project/coder>
  - Deadwood - automate as much as possible the task of updating a contributed module <http://drupal.org/project/deadwood>
  - Schema - allows modules to declare their database tables in a structured array <http://drupal.org/project/schema>
  - Backup and Migrate - simplifies the task of backing up and restoring your Drupal database [http://drupal.org/project/backup\\_migrate](http://drupal.org/project/backup_migrate)

- Drupal for Firebug - helper module for that displays Drupal debugging and SQL query information in Firebug <http://drupal.org/project/drupalforfirebug>
- Masquerade – switch users <http://drupal.org/project/masquerade>
- Administration
  - Admin – admin UL similar to D7 <http://drupal.org/project/admin>
  - Admin Menu – quick menu interface to administrative tasks [http://drupal.org/project/admin\\_menu](http://drupal.org/project/admin_menu)
  - Advanced Help - allows module developers to store their help outside the module system, in pure .html file [http://drupal.org/project/advanced\\_help](http://drupal.org/project/advanced_help)
  - Admin Message - an easy way to show short messages to users [http://drupal.org/project/admin\\_message](http://drupal.org/project/admin_message)
  - Total Control - Admin panels [http://drupal.org/project/total\\_control](http://drupal.org/project/total_control)
  - Nice Menus – expandable menus [http://drupal.org/project/nice\\_menus](http://drupal.org/project/nice_menus)
  - Custom Breadcrumbs – custom breadcrumbs for node types, views, panels etc [http://drupal.org/project/custom\\_breadcrumbs](http://drupal.org/project/custom_breadcrumbs)
- SEO
  - Nodewords –adds meta tag fields to nodes, views, panels <http://drupal.org/project/nodewords>
  - Page Title – customize the <title> tag for each node [http://drupal.org/project/page\\_title](http://drupal.org/project/page_title)
  - Site map - creates a sitemap page based on menus and taxonomy <http://drupal.org/project/sitemap>
  - Xmlsitemap - creates a xml sitemap <http://drupal.org/project/xmlsitemap>
  - Google Analytics - Adds the Google Analytics web statistics tracking system to your website [http://drupal.org/project/google\\_analytics](http://drupal.org/project/google_analytics)
  - Tagadelic - generates a page with weighted tags <http://drupal.org/project/tagadelic>
- User/Access Control
  - Access Control List (ACL) - API for other modules to create lists of users and give them access to nodes <http://drupal.org/project/acl>
  - Content Access - manage permissions for content types by role and author [http://drupal.org/project/content\\_access](http://drupal.org/project/content_access)
  - LDAP Integration - [http://drupal.org/project/ldap\\_integration](http://drupal.org/project/ldap_integration)
  - Content Profiles - user profiles as nodes [http://drupal.org/project/content\\_profile](http://drupal.org/project/content_profile)
  - Advanced Profile Kit - user profile pages [http://drupal.org/project/advanced\\_profile](http://drupal.org/project/advanced_profile)
  - Forum Access - allow you to set forums private [http://drupal.org/project/forum\\_access](http://drupal.org/project/forum_access)
  - User Points - an API for users to gain or lose points for performing certain actions on your site <http://drupal.org/project/userpoints>
- Ecommerce
  - Ubercart - e-commerce suite <http://drupal.org/project/ubercart>
  - Ubercart Discount Coupons – add coupons to use during checkout [http://drupal.org/project/uc\\_coupon](http://drupal.org/project/uc_coupon)
  - Ubercart Fee – add additional fees to products [http://drupal.org/project/uc\\_fee](http://drupal.org/project/uc_fee)
  - Ubercart Restrictions – restricts certain products and users based on location, age, etc [http://drupal.org/project/uc\\_restrictions](http://drupal.org/project/uc_restrictions)
  - Secure Pages - required pages to a SSL <http://drupal.org/project/securepages>
  - Ubercart <http://www.ubercart.org>
- Spam
  - CAPTCHA - challenge-response for web forms
  - Mollom – solution for spam by Dries <http://drupal.org/project/mollom>
  - Spam and Drupal <http://groups.drupal.org/node/26730>
- Organic Groups
  - Organic Groups - enables users to create and manage their own groups <http://drupal.org/project/og>
  - OG Block Visibility - specify that a block should be visible only within a selected group [http://drupal.org/project/og\\_block\\_visibility](http://drupal.org/project/og_block_visibility)
  - OG Menu - create and assign a menu to a specific group. [http://drupal.org/project/og\\_menu](http://drupal.org/project/og_menu)
- Miscellaneous
  - IE Unlimited CSS Loader - solves the Internet Explorer css limitation [http://drupal.org/project/unlimited\\_css](http://drupal.org/project/unlimited_css)
  - Drupal for Facebook - set of modules and themes turns Drupal into a platform for developing Facebook Applications <http://drupal.org/project/fb>
  - Voting Api – rating content <http://drupal.org/project/votingapi>
  - Fivestar – add voting widget to nodes <http://drupal.org/project/fivestar>
  - Subscriptions – enables users to subscribe <http://drupal.org/project/subscriptions>
  - Simple News – publishes and sends newsletters to lists of subscribers <http://drupal.org/project/simplenews>
  - Mime Mail – html emails <http://drupal.org/project/mimemail>

### **SCALABILITY AND WHITEHOUSE.GOV Wednesday, April 21, 2010 Frank Febraro**

<http://sf2010.drupal.org/conference/sessions/providing-scalable-infrastructure-whitehousegov>

*A good discussion of massive scaling of Drupal for whitehouse.gov*

- Contingency planning – what are the risks from code perspective (development v. production)
  - Do the load testing
- Launch prep – inception to launch in 3 months; milestones for security
- Certification process; accreditation then validation (900 page document)

- Launch locked in with 4 hours lead time
- Numbers: x00,000's unique per day; x,000,000 page views/day; 100K video stream, 15,000 forms submitted per day
- 2 data centers, replicated – staging and development in each
- RHEL with puppet, memcache, SOLR, monitoring
- Linux virtualized cloud hosting RHEL 5, hardened, SE Linux and AIDE enabled
- Puppet – data center automation
- CDN – Akamai site accelerator, Netstorage, Livestream,
- Web –all admin on non-public servers; load balance; lock down Apache
- Drupal – core 6, db replicated, full Read/Write splitting
- Cache – memcache, clustered, consistent hashing
- Search – SOLR, Nginx
- Db – MySQL Enterprise/Innodb, RAM file system for temp tables.
- Replication – two tiers Mater/master; passive is for slave replication
- Monitoring – MySQL Enterprise monitor, Nagios, Cacti
- Replication Monitoring – custom scripts, monitor all slaves, remove slave if mysql or replication down, re-initialize slave and add back, manage replication hierarchy
- Environmental Sync – sync assets to Akamai, NetStorage; sync virtual host; replicate search index; replicate db to db servers
- Hardware – quickly scale horizontally; Puppet puts new hardware in place in minutes
- Data – core content; high volume web form submission; allow for quick partial recovery (segregate data)
- Dev Process – many developers; branch per issue; branch per release; subversion; GIT (eventually fulltime)
- Release Process – branch/integration sites, one deployment per week (at least 1); JIRA, Fisheye, Crucible
- Eg, Save Awards – use five\_star module; visitor records, table query, bulk import and scaling thru Drush; mobile and iPhone version; HTML 5 version;
- Next... scale user authentication, user-generated context, 2 million web forms at launch→ MongoDB

### **AEGIR HOSTING SERVICE Wednesday, April 21, 2010**

<http://sf2010.drupal.org/conference/sessions/aegir-hosting-system-one-drupal-rule-them-all>

*A demo of Aegir – for hosting solutions. A scattered presentation. Presenters (developers) not impressive.*

- This was mostly a demo. See the URL above to view Demo.
- I don't think Aegir is for us!

## **Eva's notes <egrabins@uwaterloo.ca>**

### **SUCCESSFUL DRUPAL PROJECT EXECUTION –APRIL 19 2010, Presenters: Michael Morris and David Leonard**

- big complex drupal projects; how to work w/ stkhlders; executing difficult drupal projects; defining scope and reqs; addressing common situations
- the non technical issues on drupal projects
- how to work w/ stkhldrs. Communication is very important.
- PM techniques and project execution.
- Defining scope and reqs. Need really good reqs definition – a must.
- Solving issues with content etc that common with drupal implementations.
- Phase 2 technology is the company. Lot of multi site. About 80 sites or so. Work with big orgs. Contribute to community. Have modules and support. Have installation profiles given to community.
- Stakhldrs. Give into to drupal – first thing to do. Got to know the terms. Kick off meetings starting w/ terms – important.
- Got to explain open source. Most stkhlders don't understand what this means. Understand the free part. Don't really understand open source community model.
- Some people do know about drupal. And think or know they want drupal. A little drupal knowledge is dangerous. Module hype is dangerous.
- Drupal is not magic. It is platform and tool. Use like any other tech. Must have good practices and good techniques. Need to communicate it.
- Set expectation properly. Need to set time expectations. Can't just knock out drupal websites quickly. Need to set time expectations. At least 3-4 months to get sites into drupal.
- content needs to be ready ... this can stall projects.
- Use project portal – like basecamp or something similar – for project communications.
- Have project status reports. Really important.
- Have regular meetings.
- Communicate regularly.
- Don't toss out great pm practices b/c using drupal.

- Drupal hard to grasp for many. Users have trouble with views for example. Takes technical know how to navigate drupal for many things want to do.
- Train stkhldrs. In person training is the best. Administration guides very useful. Too much documentation can get stale. Need the right mix. Have a starter guide – welcome to your cms. They use screencasts – take a lot of time (we're not lullabot so not like that) – focus on just small task focussed snippets – easier to create and manage.
- Do stkhldr testing. Must test. No substitute for stkhldr testing.
- Is the taxonomy meaningful to stakeholders? Test for this.
- Does it meet their needs? Must, must test. Not about being buggy. About working for the stkhlders.
- Have to make sure design theme looks great – make sure design turned into theme correctly.
- Absolutely must involve your stakeholders. Help them, give them tools. Got to make the stkhldrs accountable – they need to work too. The stakeholders have to work hard with you.
- Executing difficult projects.
- The Drupal learning curve is very. Drupal seems easy at first. Then you hit a wall. Need to understand this. Very easy to get too lulled into how easy it seems. Be prepared to make lots of mistakes – and learn.
- Right project team. Need the right team. Need soft skills – need pm, req analysts. Not just developers. Pm involved through entire project – very important. Risk mitigation, status updates – needed to communicate with stakeholders.
- Need an analyst – title could be solutions architect. Understand problem and turn into workable solution. Must do user experience, wireframes, defining content types.
- Designers and web producers -
- developers – more than 3 or 4 developers makes it too difficult – they tend to bang into each other – need primary developer.
- Testing team would be great. Or at least have testing.
- Encourage role flexibility. Have overlap in roles.
- Being agile and managing scope. Agile shop. Different forms of it. Most important working in iterations. Define what you can upfront, but can do later in chunks. Be firm on scope but flexible on how to implement features.
- Open backlog. Be transparent. Helps everyone understand project scope.
- Interteam communication. Use jira as their task tool.
- Realtime communication – this is important. Especially if your team is all over the world.
- The pm need not be involved in all conversations – just informed.
- Insist on quality. This is technical testing. Create good quality product.
- Soft skills are really important to Drupal implementation. Really important.
- Scope and reqs. Very very important. What are the sites goals, are the audiences tech savvy, how do editors contribute content, what is focus of the content, etc.
- Reqs toolbox. Sitemap, wireframes, specs, system diagrams, design comps etc.
- Sitemap - great communication tool. Yes moving away from hierarchy organization, but still good first point of discussion.
- Whiteboarding. Pm and designer lead/help.
- Then end up with wireframes. Most most valuable way to document a site and it's functionality.
- Don't neglect the details. Developers tend to just run with it. Get it described right up front. Will save time later – big time burner if don't get details upfront.
- The website is the easy stuff. Working with other stuff is important to consider. Make sure systems can talk with another. This takes so much time.
- Document the specs. Formal reqs document. Stakeholders often require it. Can also use wiki to define the requirements. What way works best for your team. Must have documentation.
- Some won't get it until see design comp. This is often the ahh-ha moment. Make sure they understand previous documentation before you get to this stage.
- Scope and reqs. Lots to account for. Lots of questions for drupal projects. Proven techniques. Classical stuff that works well in drupal space. Think about the whole system – not just the website – the whole system has to work.
- Now onto drupal specific stuff ...
- common challenges ....
- use good habits – replace bad ones. Help stkhldrs understand drupal diff beast. Spend time upfront watching what web maintainers do – don't want surprises later. Large orgs have lots of different roles and maintainers.
- Structure content properly. Straightforward content types. Complex content types – that can relate to others. Define your content types upfront. Really got to do upfront.
- Permissions. Who can create post. Etc.
- Use taxonomy effectively. There is the traditional sense. Very easy. Great for major sections of site – topic based nav, targeting ads to sections. The callay (?) service – semantic data for your content – converted to drupal taxonomy terms – automated tagging of content – certain vocabs provided may not work for you. May want to use node references instead. Or maybe want both.
- Admin interface. Use admin interface. Make sure it's a clean admin interface.
- Help text. Extremely extremely important. Helps users understand what fields should be used for. Avoid jargon.
- Image fields. Image cache can help. Do leg work on your image sizes. Very important.
- Everyone loves wysiwyg. Great way for content authors. Not intended to be to manipulate html. Make nice editor – use css for realistic preview. Should really do this. Text editor. Start w/ a very small tool bar in text editor.
- Allow editors to curate and innovate. They may be used to flexibility. Let them have space for customization – just simple basic customization.

- Editorial workflows. Most recent content created may not want up on site first – remember need to be able to define what order content needs to be published (which may not be the same order that it is created in)
- represent content relationships correctly. can use node references or taxonomy. Node specific coupling relationships. Taxonomy more traditional way to create relationships in drupal – filtering, tagging – not easy to expose in other places.
- Differentiate between authors and users. Big, big tip!!! create author node type – used as an example – for multiple authors.
- Use apache solr for search. Great functionality out of the box. Sort and filter by facets – can be node type or taxonomy. Powerful with callay (sp?).
- Roles and permissions. Pay attn. Don't have job function tied to role – big problem. Way too many roles. 8 roles is bad – way too many. Keep roles to minimum.
- Scale back permissions. Limit administration permissions. Not trying to hold in. Can maybe grant permissions gradually.
- Content migration. Big piece of project. What are they doing on regular basis. Early content type definition important. Wireframes help. Migrate and table wizard could help – as could custom scripts for scraping.
- Mitigate your project risks.
- Drupal projects have lots of challenges to solve. Use drupal features and functionality the right way. Never assume the module will satisfy requirements. Meet needs of editors and writers – important. Solve problems with right techniques.
- Summary ...
- the harder the bigger the drupal project , more to tackle. Good module developer not enough. Need different talents.
- Questions ....
- how much discovery upfront. Need to stop at some point. Ask lots of questions. More information means better project.
- Estimates a tough think to do for projects. Learn from previous project.
- Documentation. Not too many formal templates. Some is good – depends on size of project. These are pm templates. Depends on the team. Sometime use formal templates. Sometimes just whiteboards. Sometimes just wiki. Whatever works for the project team.
- Firm of scope flexible on features. Have to bound the problem. Fixed price or time and materials – when economy bad end up doing fixed price. Bound the problem. Important. Sorry out of scope.
- Requirements gathering. Should be actual deliverable. So should sitemap. So should wireframes. Should meet to go over with client together.
- So many assume drupal is easy.
- Share what you know. Educate about drupal and managing drupal projects – large drupal projects.
- How deal with environments and quality. Complex problem. Development, staging and testing. Depend on project. More process in place would be good. Testing is important. Use tracking tool for tasking. Works really well on large projects.
- How much training. Minimum 2 big training sessions. Each 2-3 hours. 3 or 4 working days to prepare (noting have done it before and have starting points). Depends also on how much site owners will actually do.
- Learn about subversion.
- Internal testing. Alpha release. Do training. And do testing at this point. Get people involved in using site as soon as is possible. Don't assume what they want. Get and keep them involved.
- Detailed reqs gathering. Rfp process? Can't get too detailed in rfps always. Build analysis and reqs into phase right upfront in project.
- Prioritize trade offs. Prioritize, prioritize, prioritize. Needs, tasks, etc. Note that priorities can change – as stkhldr learn more about drupal.
- Dealing with a lead. Set up a preliminary meeting. Determine high level scope to determine if it is something within your business scope.
- Have standard timing and placement for status reports. Let stkhlds know what will be working on next week. Prioritize.

## **THEME PREPROCESS FUNCTIONS –April 19 2010, Presenter – Carl Wiedeman**

- demo focused session
- preprocessing in theory
- relevant to d6 and d7 – theming gets more complicated for d7; easier to start with d6 and then move to d7
- used to solve common theme dilemmas
- preprocess makes content flexible; makes theming dynamic; keeps templates clean – leave php out of these (out of template files)
- preprocess in practice
- admin\_menu, devel and devel themer are MUST haves - modules
- theme.inc file. Can find preprocess functions sitting in this file. Look at page.tpl.php file – sometimes knowing what in here not enough. Got to go to theme.inc file to see what's going on. There is an array of variables in the .inc file. Whatever you do do NOT change the theme.inc file – this is part of drupal core and you will mess up drupal.

- The template.php file. This is where you put your own preprocess functions in. Each tpl.php file has it's own variables.
- devel module essential to preprocess functions. Use dpm() to dump variables. Don't forget to clear theme cache – flush the cache (several ways to do this). \$vars is an array. Devel lets you see what's in this array. Some things are logical here – somethings not so clear in terms of names given.
- Parameters from preprocess functions get sent to .tpl files.
- Example of dpm(\$vars) for node.
- Where we can change variables. Use case – don't want it to say 'submitted by'. Find submitted node. Well want the default for submitted to be different. \$vars['submitted'] = 'test'. Test first to make sure works. Then move on to what want to do. Added display of time when content created – using a little bit of php.
- Drupal API is a crucial resource. Every function in core is listed here.
- Can add your own variable. Example creating an array of greetings. Write the php you need. Now remember to print in the relevant .tpl file!
- Targeting a specific node. Use dpm() to do this.
- Know what vars are already available in theme.inc file – important. Start by knowing what variables are available to you – before creating new variables.
- Got to know unix command lines – no way around it. The way to quickly see files you have as part of your drupal installation/set-up
- Reminder again about clearing/flusing cache.
- Taxonomy example. Creating a vocabulary – sports vocabulary. Create story and select term – and do it again. Client decides it would be cool if theme changes with taxonomy term. Fun! Use dpm(\$vars) to see what's there already. See template files page-taxonomy, page-taxonomy-term, etc – but maybe better way to do this. Go to the node!!! open up via devel module to learn about node that you will want formatted with a specific subtheme. Should have access to the node object. Oops got to go to page. Zen is a good place to start. Switching to Zen because good to work with. Zen is really useful way to sub-theme content. Creating zen sub-theme. Checking out template.php file. The node object is there!!! Yay!!! Zen is your friend – really start there. Start with node object and the taxonomy object – starting with Zen. Look at the taxonomy object within the node object!!!! Is possible but demo not working.
- Umm, remember to clear cache. Mentioned over and over again.
- Oops, presenter didn't practice demos before session
- CCK is something you got to use.
- Firebug is essential for drupal theme development.

## DRUPAL 7 THEMING – APRIL 20, 2010

- Theming will be a big change in d7
- What's similar: not complete revamp; basic concepts the same; CCK becomes Fields and part of core; now have image fields; Image Styles replaces Image Cache
- Lots of themes approved for d6 suck – about 75% of them; in d7 crappy themes will be gone; usability was an issue with the crappy themes – d7 is taking usability more seriously; garland, stark and seven will now be the default themes; there will be clear distinction between an admin theme and site theme – b/c of usability
- The Zen themer's starter kit available for d7
- The page.tpl will be a cleaner, nicer starting point
- Name changes: primary/secondary => main/secondary; clear-block => clearfix; \$left/\$right => sidebar-first/sidebar-second (for multilingual support – reading right to left)
- .phptemplate\_prefix GONE – do not use in d6; there is a bug w/ this in d6; stop using in d6
- Theme functions now have only one parameter (an array) in d7
- Easier to create more regions in d7 – means less hard coding; there are now blocks for things like \$search-box, \$mission, etc.
- Content is now a block in d7
- 'Template suggestions' gone in d7 – replaced with 'theme hook suggestions' ... already part of Views (feature in Views in d6)
- node - - article << way to handle just articles
- jQuery UI now in core in d7
- .html.tpl.php file now exists in d7; certain bits in the page.tpl now in the html.tpl
- d7 more accessibility compliant (w3c, 508)
- Granular theming. Lots of excitement about this. This is why the html.tpl file now exists.
- Reminder by presenter that use cases are important
- hook\_page\_alter() new in d7
- Don't have to be a super PHP ninja in d7 to do theming
- Check out the bartek theme
- In d6 Zen is a good start; in d7 all themes will be a good start

## **CCK DEMYSTIFIED –APRIL 19, 2010, Presenter: Doug Vann**

- dougvann.com
- going away in 7 – will be fields and will be in core
- fresh acquia install – starting with this – all you need is there – using the acquia interface
- dualconsulting.com – thank you to
- cms quick start – installation profiles – this is a website
- recommend to start with acquia drupal
- drupal comes w/ page and story – one promotes to front page other one doesn't – all it is title and body and this often just doesn't cut it.
- Job posting example. Example of custom content type. Get stuck with default page and story content types. CCK to the rescue.
- polls, books, etc. All content types. Just an aside.
- Modules page. Use content module – enable. Email is useful module too – an aside. Looked at a bunch of modules that are part of or relate to CCK.
- Take a page or story and make more granular options – for job posting. Now have manage fields option w/ CCK.
- Demo of creating content type w/ CCK. Machine name all lower case with underscores. Definitely use the description field to describe what the content type is about, what it should include. Use the explanation section to help users. Review of default options (e.g. Published, promoted to front page, etc.). Comment setting – no thank you for job posting. Remember to save content type. Now go to manage fields. Demo how to add a new field. Allowed values – just list them here – follow the instructions in CCK. Wow have new content type.
- Great clear explanations!!!
- Always have use cases for your content types.
- There's more ... let's talk money. Got to add salary range. Remember to turn on the number module!!! an integer defined as text field. Remember to hit save. Reminder, know your use cases!!! note that commas don't work – and only you can remove commas. Be sure that you dept selection is a drop-down or radio button b/c must be selected. Got to turn on widget options!!!! - this is a module that must be enabled!!!! Select list sounds good for this.
- Create content -> job posting.
- Path module – gives semantic names to our paths. Recommended. Now have URL path settings option – sweet. Lets you put in the url.
- Manage display fields. Can move things around easily with handles.
- Title is not field b/c default – same with body. But can modify other fields. Can do stuff with the display field tabs – adding commas to salary, etc. This about how field data displayed – not about how data entered – but about how the data is displayed.
- Use theming tricks to make content types appear like you would like.
- You don't have to use the teaser. If you don't fill it out, it won't be rendered. Teaser is first 600 characters of post. Teasers go to front page rather than full body text.
- Tagging can get word. Then you have synonyms – this is all about taxonomy.
- Adding images. Image cache – really cool – crop, scale etc for web use – sweet. Default in drupal reviewed. Where will the image be stored? Create unique path to store such images – tip. Make sure you enable alt text – really a must. Default image is fun – don't use it much.
- Can you enforce presence of alt tags. Don't know. Haven't thought about.
- Use image cache – very important!!!
- Each department has own web page. How about a node reference. Select all department nodes for particular department.
- The media field allows videos etc. There are plugins -e.g. For screencasts. See lullabot resources for such stuff.
- Creating a collection of fields – can use field groups.
- There all all sorts of CCK add on modules – check what's out there before venturing off on your own.
- Theming forms in drupal is an art.
- Can use mutli forms to help break up forms into multiple pages. Think i got this multi forms name right?
- Sharing fields – you may not want to share fields. You can share fields but may not want to.
- Big note -static site to drupal or similar. Thing about pages. What type of content types do you really have? Really think about this!!! start thinking about content types need to pull off project. What is the user going to need??? Think upfront about what will need to be included upfront for content type – or will be painful later. Look at web through eyes of cck.

## **ALL ABOUT FEEDS MODULE FOR DRUPAL - April 19, 2010**

- Look at feeds section on drupal.org
- Feeds is a drupal module
- Allows to enter feed URL

- OPML import – showed how feed subscriptions imported
- Node import – just mentioned – not much more about it
- Focused on participants who already use aggregators and feeds
- Hard time hearing presenter – microphone shy
- Leech – aggregator 2 not good to work with – according to presenter
- Discussion of FeedAPI – just talking about the history of (not anything more)
- Historically not much interest in aggregation (in drupal community). Also drupal not that great in terms of available aggregator functionality. Great expectations w/ d7
- Aggregator for d7: pluggable parsers; pluggable processors; mapping API; multiple configuration of importers
- Also goals of: suitable for aggregation and import; more than http get; clean extensibility; exportable configuration. The goals were not moving along well so interested in getting goals moving along.
- Feeds – complete rewrite of how aggregation handled.
- Importer overview slides. Core of feeds is an importer. Handles the import, holds plugins. Slide showing fetcher, parser, processor – creation, parsing, consumption (batch process) – http request, rss handling, save node (process example)
- Content types and their relations to nodes is key.
- Why would you attach feed to node? Well this is a good way to handle aggregation
- Aggregator in drupal core is an importer. Says aggregation not really handled that way anymore – that is, not really handled like within drupal core anymore.
- Showed feeds module UI.
- Feeds module is still alpha. Nowhere near ready yet.
- How to expand what is there. Feeds plugin API. Discussion about using the fetcher, parser and processor.
- Also mentioned Feeds mapping API. Allows mapping to taxonomy. Allows mapping to CCK field.
- Discussion of taxonomy mapper. Showing PHP for this.
- Some mappers are more complex. Really need to be based on use cases.
- Typical use cases ...
- managing news. News portal. Aggregator that maps news to maps (geographical maps). Shows hot spots on global map. Geo-tagging of news items is what's being done. Syndication to manage news.
- Open data kit (ODK). Little framework using java (?) – creating stuff for google android. Showed mobile device – captures photo and geo-location of photo – pushes information to a Drupal site.
- PubSubHubbub. Problem in aggregation – do lots of cycles of asking for feeds. Better way to handle this. Helps with scaling for aggregation. Best way to deal with scaling of aggregation. Showing Feeds integration with pubsubhubbub. Google involved with pubsubhubbub project.
- Showed how to create a subscription. This is via using the Feeds module.
- Where going ...
- This module will never be finished – always something to do. Will be shaped by community.
- What hot ...
- dynamic update scheduling
- parser level batching
- views based parsers
- configurable (extensible) xml parsers
- ui improvements

## PHP FOR DESIGNERS – APRIL 19, 2010 Presenter: Emma

- Very basics of php; need to know php for design
- Emma writer of front end drupal book; php for drupal book; theming drupal book
- must, must know php – not an option
- variables - think square dancers; variables allow placing content and blocks into website
- variables have to get printed. Have a bunch of tpl files.

### Musts:

- /modules (know about this directory)
- download zen theme (start here)
- root candy for D6 (good resource)
- \$language->language – talking about \$node object
- find variables at /modules/system/page.tpl.php – do not edit files here EVER; can copy over to your theme but do NOT edit here
- must use devel module and themer module – not an option – really a must
- \$node object
- find what you need and alter it – this is pretty much what you do with theming (and the related PHP)

- conditionals (if-then statements); most common use is collapsing regions when there is no content to go into the regions
- ( equivalent to action word – something good is going to happen! Conditional statements or functions – this is what round brackets indicate.
- user\_is\_logged\_in () - example function
- The t is for translate in php – when see before round brackets
- functions have parameters stuffed into them theme('links', \$primary\_links, array('class' => 'links primary-links')); what you will manipulate, then array, then extra stuff you want to add to variable before rendered or output to page; want to add class to css for handling links – that's what this is used for; useful to know how to add additional items to arrays
- theme is a specific drupal function
- know your theme functions – or where to find them
- single quotes – these are literal – no conversion happens – just totally literal
- double quotes – convert into a value (convert variable into value)
- commas separate parameters
- decide that php is easy – just go ahead and do it
- imagine what you want to happen; figure out where needs to happen; write in some comments so you know what's going on; then figure out what php you need to make it happen; then fill in the actual code ... always know what you need to have happen first – that's the first step!!! php becomes about what you want to have happen
- breadcrumbs suck in core; need breadcrumb module
- no problem with over commenting – good way to start ... better than not commenting enough
- example snippet for node.tpl.php
- \$ = variable
  - ➔ arrow means object – have properties
- book is a content type; books about hierarchies – have parents and children; not parent (\$node ->parent!=0)
- number of parameters is equal to number of commas – easy thing to remember
- PHP snippet example
- Check out drupal.org b/c the snippet you need may already be there!
- Really good and fun introduction to php for designers (theming)
- preprocess functions – allow you to make your own lunch! :) alter the content of variables that were prepared by drupal and its modules; create variables that weren't assembled by drupal and its modules; create fun functions to do things you want
- template.php file.

Note – didn't discuss if-then versus if-else

## QUESTIONS

- Storyboard it first; know what you want to do with the php and then do it
- Copy the files you want to alter into your theme – don't bring over any extras
- && means both have to be true
- [ the way of getting into an array
- taxonomy term off of content type – section in front end drupal about this
- Zen framework is a really really good starting point
- PHP for women on freenode – check it out; php women channel
- eclipse is pretty good; kumodo; she likes vim; edit plus ... there are a bunch of different ones

## VIEWS DEMYSTIFIED – Part 1 and 2 - APRIL 20 2010, Presenters: Rain Brew, Doug Vann

### PART 1

- must use Views
- need CCK, Taxonomy, Token, Link, Date, FileField, ImageField, ImageCache
- can't really do Views without CCK
- need to know about how core Drupal taxonomy works
- Token allows replacements (e.g. Of URLs)
- an image gallery is a very common use of Views; along with FileField, ImageField and ImageCache
- take the time to really learn Views – it takes time
- imagine it first, then build it
- it's just a PHP and SQL query builder

- can use Views for RSS; Drupal default for RSS not great – limited; can customize RSS feeds with Views
- can use Views to help create a special front page – distinct from other pages of your site
- can link searches to taxonomy using Views
- can create sorted lists by taxonomy using Views
- rotating view example – display field from content type – for rotating image on front page; can create a Views slide show that rotates through images or videos
- Views is about data not appearance – need CSS to make things look pretty
- Views are often used to create FAQs pages – accordion implementation of FAQs
- recommend admin menu module – a must have – set up from the start
- Views controls how things display without touching the theme
- some Views appear when you enable/install modules
- italics signify whether what is in a view is custom or default
- can filter by node type
- yellow background means that that item for a View has not been saved
- sort criteria – the order that data appears
- the starting place is filters
- tag your view – very, very important
- the published filter is important
- note the taxonomy filter in Views
- fields in custom content types can be used to filter in Views
- only super user (user 1) can use Views
- limit roles to 3 whenever possible
- Views OR – good module – but still in the red – can break your site so test it first
- 2<sup>nd</sup> thing after filter is sorting
- always use 'Hide if empty'
- can put basic formatting onto field in Views – h3 text /h3
- give each page a URL – important; be careful with URL naming conventions; Views will have problems if you have 2 paths that are the same

## WARNINGS

- italics overrides defaults
- remember to save
- include 'Published – yes'
- don't add extra stuff to your Views – it will slow down your site

## PART 2

- start with filters always
- set published to yes
- filter is your db filter
- give a meaningful name to your View
- italicized means customized
- discussed attachments, how to use, and settings
- trim field example – strips html tags (teaser w/ 140 character limit – example)
- use 'hide if empty'
- discussed offset option. If put 1 in here then this item will not display 1<sup>st</sup> item. Awesome for having teasers of older posts below first full feature post (blog post)
- when creating a feed always put .xml extension; should end with "rss.xml" always. Add this to the path when creating a feed display. Important.
- Use taxonomy to create a menu within a block using Views. Example of recipes: salads; appetizers; sandwiches; etc. For mini menus 'in' pages.
- Use attachments versus more blocks where possible. Better for performance optimization.
- Taxonomy terms. Have x # of taxonomy terms. Goof for RSS feeds. Sort by taxonomy term. Shows taxonomy term associated w/ each node. You can nest your criteria.
- Node example. How to show sponsor logo only on certain web pages. Can use your page URLs for this – there is a reason why the naming conventions for URLs/paths are very important. Use path associated with relevant nodes to display a sponsor logo on only certain relevant pages.

- In terms of theming views, understand how the theming layer works. Note that you can also put classes in views – using the text field to do this. Can apply a CSS class based on a taxonomy term. If content classified as appetizers, apply class for green content. Can have it apply to a parent in the taxonomy and the parent's children.
- Carefully tag your views – and remember to tag your views.

## **JQUERY FOR DESIGN AND THEMING – APRIL 20 2010, Presenter: Bevan Rudge**

- distinguish between designers and themers
- sample code at [drupal.org/Project/Q4Dat](http://drupal.org/Project/Q4Dat)
- library to extend JS
- context is web pages
- selector engine (like CSS) – selector engine is Sizzle
- a library for manipulating DOM
- .info file – where to add reference to JS file
- always do 'hello world' test to get started
- be careful about names you choose for files, classes b/c – make them unique to avoid clashes
- presentation using acquia Drupal and the devel and theme developer modules
- make sure JS is enabled on your browser – note on turning off JS in browser to test for accessibility (508 compliance)
- [visualjQuery.com](http://visualjQuery.com) – really important resource; can find functions here w/examples; a bit out of date but still really good resource – [jQuery.com](http://jQuery.com) is not as good b/c harder to find stuff
- pay attention to usability – you can do stuff that simply not usable
- Firebug is a must
- really know how keyboard navigation works to ensure accessibility and usability
- textmate is a tool you can use
- watch your syntax – it is different than PHP
- always use ; at the end of your lines – this is best practice
- check out jQuery UI; don't load all – just the bits you need
- remember the README.txt file
- copy good code that works already and take it from there

## **SEMANTIC WEB AND DRUPAL – APRIL 21 2010**

- rdf is focus of presentation.
- Semantic web.
- benefits ... exchange information across programs and sites; content reuse; search engines can display more relevant information; data mashers – can find unexpected trends and correlations
- keywords like machine understandable, rdf, linked data, etc.
- Machine understandable. Right now data is human understandable. Understanding content beyond just text data. Machine understandable context. Context. Understand that this is name, etc. That is, that machine understands the context of the content/data.
- Rdf. Resource description framework. Breakdown into resources – a resource is a named thing. Needs a unique name. A url is a unique name. Long and clunky. Can shorten url down. Compact url.
- Resource could be document, company, person, etc. Talking about what a type is. A person is a type. Can talk about how resources relate to one another. There are types of resources.
- Federated dataset. Giant global graph (db).
- Sparql – like sql specific for semantic web.
- Rdf – use to link data together.
- Why drupal - structured data -data structured in similar way as for semantic web use
- diffs: drupal structure hidden in db - rdf structure exposes data on the page; drupal field names specific to drupal site - rdf terms universally and explicitly defined.
- History of semantics in drupal.
- Started looking at this in about 2000. getting rdf into drupal – the rdf module – hasn't worked – lots working on it but lacked api all modules could use. Raised money to get developers together to work on rdf. List of issues with rdf for d7.
- What is rdf and rdfa. Rdf around for while – describes semantics of data – represented as graph – about expressing an idea – use english or other to civilize the data – rdfa is how you put in a web page and the a stands for attributes

- microformats versus rdfa
- rdf and rdfa. It's a standard. W3c. Bbc, nytimes, reuters, fb. Google and yahoo. Don't repeat yourself (dry).
- Bits of semantics in your html code.
- Many vocabularies to put into your pages.
- rdfa helps with ecommerce.
- rdfa helps with seo
- Node, comment, user, file, term, vocabulary. These are entity types. Machines and crawlers will understand these types. Attributes: title, all fields, date (created, updated) comment count, reply of comment, creator.
- Cool URIs. All entities have unique uri. Node, comment, user, taxonomy term.
- Foaf, sioc, skos, dc are out of the box.
- All users have WebID. For use all across the web. Secure authentication on the web.
- Graph of blog post. Boxes and arrows all over the place. Rdf data graph.
- Rdf mapping ui for site admins to customize mappings. Module.

## WEB DESIGN PM FOR DRUPAL - APRIL 21 2010

### PHASE 1

#### \*\*\* Discovery Work and Deliverables\*\*\*

- Use cases: understand what visitors to the site need to do/find
- Client survey: understand who your stakeholders are and what they want out of the site
- Competitive analysis: understand how visitors expect sites in vertical to behave
- Content review – really a content analysis/audit – to identify content types early on

### PHASE 2

#### \*\*\* User Interface Work and Deliverables \*\*\*

- Site map: translates into the navigation menus in Drupal
- Wireframes: use grey models: translates into regions in Drupal
- Basic prototype (low fidelity prototype): use Garland in Drupal to implement basic site to test web IA usability – based on content types, site map and wireframes; have some preliminary sample content created to help show how site functions; implement more basic Drupal configuration at this point – start with content types, etc.

### PHASE 3

#### \*\*\* Graphics (Appearance) Design and Drupal Configuration \*\*\*

- Appearance design: design appearance of website (typically only one design that gets tweaked)
- Custom functionality: create customized functionality (do as early on as possible to mitigate project schedule risk)
- Appearance design iteration: about 2 or 3 iterations is typical
- Drupal theme: create the Drupal theme (once have sign off on appearance design and once the standard non-theming functionality in Drupal is implemented)
- Drupal website: bring it all together and do all that required for launch

### NOTES

- Start high risk functionality (the custom stuff first) – makes project run smoother and faster
- Client survey key to understanding what your stakeholders want out of the site
- Site maps are still the best way for clients to conceptualize their sites IA
- Site map can be simple or complex depending on project and on client – it has to be presented in a way that the client best understands – some like boxes with arrows (like org chart) and some understand simple lists in a Word document better
- Some clients have trouble understanding web design until the visual component is presented
- Keep format of deliverables as simple as possible to meet client needs and to help clients understand the process

- There is some overlap with phase 3 and the end of phase 2; appearance design often gets started just after the wireframes are approved by the client but not before – we may have some concepts in mind but we have to know that they will work before we put time into the appearance design; note that good to start the appearance design at this point because you will have learned a lot about what the client likes and dislikes by this time in the project; note that implementing customized Drupal functionality often starts during phase 2 – this is recommended to mitigate risk with respect to project schedule
- You want to do the usability testing at the basic prototype stage – when the theme isn't affected; otherwise it becomes a lot of work to tweak the design/theme; have your wireframes and your low-fidelity prototype approved before you theme in Drupal

**WEB TYPOGRAPHY: TAKE WEB DESIGN FROM GOOD TO GREAT –APRIL 21 2010, Presenter: Samantha Warren**

- focus on understanding web typography
- samatha warren – senior web designer at phase2 technology – badassideas.com – twitter @Samanthatoy
- open source font movement
- think about texture and typography – can make this happen on the screen
- think about how letter create pattern and sense of movement
- a lot of the best typography is on alcoholic beverages
- brief mention of sifr
- surf typography – the feeling of typography
- truvia; ojectified (movie) - examples
- use of white space – will help emphasize a word
- print designers don't often understand what limitations the web imposes – but there are lots of things you can do on the web (like print historically, can have 2 fonts and can do beautiful design – about placement, colour, size, etc.)
- see john fan website – pushing fonts on the web – all html and css
- poster exhibit
- web design habit – look outside of web for inspiration and bring into design
- firefox – example
- web imposes constraints – find the freedom in the web design space
- evaluate the situation – who are the actual site visitors – know what browsers they are using
- know the technical and brand requirements – make sure these are understood
- what is predominant os and browser of targeted user
- is the site content heavy
- does consistency across browsers matter
- is there a budget of fonts – fonts cost money – really good fonts cost money – clients often don't get that – may have to find free font to push the boundaries with
- legibility – important important important – thing about how users will digest information – stroke width – shape exaggeration – proportion – x height
- helvetica is highly legible – good body copy font
- pay attention to colour contrast – very important to legibility
- RefreshDC – check out website – groups that meet about web design (like drupal groups)
- display type – doesn't change – can use neat font here
- body font and heading font helvetica – quick legibility
- what is the emotion of your site? Think about this. Is is modern, is it artsy? Like choosing a pair of shoes. Shoes go with personalities.
- Tungsten – readable font – can be trendy
- archer – trendy font – like converse all-stars – friendly approachable typeface – but trendy – maybe just because they are good fonts – it's the martha stewart font – the approachable font – both a bit formal and also approachable – good for government websites – common on wedding invitations
- go by feeling not trends – make feeling guide your choice not trends – think about concept you want to communicate – go shoe shopping
- example site – selection of fonts – font shop etc.

- how to execute – font stacks, sifr widely used – sifr is flash replacement – these are not practical – sifr requires flash plugin
- microsoft – created program to create fonts – verdana, georgia, arial etc. - created fonts for folks to use; core web fonts
- cufon – method for consistency across browsers – not selectable – no plug ins
- important to understand end user licence agreements
- gotham has end user licence agreement – they do allow sifr – but not other uses
- @font-face – around for a while – recently popular – open to fonts with eulas
- no agreement really about file format – microsoft format, firefox format – fonts world is a resource
- font squirrel – fantastic site – great place to look for fonts – great for cost effective, free fonts
- font services – they take care of legal aspects around fonts – have set pricing structures for pretty small fee – typekit – fontdeck – k\* - font services also generate the fonts you need for the web in the formats you need – you are limited to the fonts they have
- css3 and type – css3 not supported by IE
- andy clark – 24 ways article – designing in the browser
- text shadow – beer camp website – all css and html
- multiple columns – lots of people don't like columns -column with 300 px – column gap 30 px – but columns really work well for many sites
- rotate. The 30deg example. Let's be friends example.
- Making your own font. Yourfonts.com – not high quality – for the screen and not for print. If you are serious then fontlab.com is a vector tool.
- Open source font movement. Get involved. Example raleway font.

## AGILE PM FOR DRUPAL PROJECTS –APRIL 21 2010

- waterfall can work really well for drupal projects – and has worked well
- can mix waterfall and agile – depending on sub-teams for example
- canban process – not really recommended for development – scrum better
- recommend getting clients on retainer because drupal implementations are often agile – oh the client wants this too
- agile process let's client prioritize things after they understand a bit more about the product
- hours and money – got to really get experience and then take it from there
- user stories are useful and important – averages out to be about \$500 per user story point – this is derived based on significant experience – the points system for user stories to determine dollars doesn't work ... really decouple this; points is really just used for iteration planning
- determine whether the client or someone part of your organization is the product owner; having client as product owner requires client that heavily involved and available to project; know your client and whether the client should be product owner or your organization should be product owner
- 4kitchens – jira – recommends jira – this is issue tracking tool – really for scrum – tracks scrum points
- ensuring personnel available for sprints – required for sprints to work.
- Greenhopper – lots of folks excited about this – but doesn't work if the client doesn't enter information and keep their information up to date – free for non profits
- balsamic mockups – can integrate with jira
- sprints 2 weeks long – 1 week not long enough – 3 weeks team members procrastinate
- get clients to prioritize user stories – this is a must – prioritizes what you are to implement
- daily scrums per project
- *how does this work with dispersed project teams? How does this work when personnel (team members) report to more than one manager and you don't allocate team members time*
- scrum is a framework. Collaboration is the focus. If they want it they want it ... if they need it they need it. It's about getting the client to prioritize their needs and wants. Do baseline requirements. Good when developers talk with end users.
- pmp – a different approach to project management.
- *Where does scrum fall apart???* need to understand this too.
- Understand the client. Agile may not work. Depends on client. Can use agile in pieces of project rather than whole project. Client dependent thing really.
- Scope, deadline, cost – which do you want fixed. Pick.
- Drupal – features don't cost money but customization does. Set expectations. Clients don't understand why this feature fast (b/c comes with drupal) and why this customization is not.
- Have to keep the client informed of the process. This is important.

- Assess your client. What is their availability? Do they want to be heavily involved or not. Need to modify your process for the client.
- Sometimes have to use pm methods and tools that the client comfortable with. If it's excel spreadsheet, then it's excel spreadsheet
- waterfall good for predictive projects – or predictive aspects of projects
- note that expectations of IT staff and communications staff tend to be different – understand if one or both are your stkholders
- sometimes don't have to educate the clients on drupal – they just want a website that works – that's all
- in the end it is about a site that's useful for the end user – i.e. The website visitor
- usage centric design – user stories – then wireframes
- are you selling website or are you selling drupal – you are often selling website – often clients don't care at all what the technology is – they just want what works
- a lot of clients have business plans – know if your client has a business plan and understand it
- some clients want and need to know about drupal – e.g. they want to be able to expand and maintain after the project
- sometimes clients just want consulting on best practices – recognize this
- scrum problem is that see trees not forest. Need to have in context of objectives – need to know wider objectives.
- Recommend retainer and not fixed bid.
- Drupal easy to start and give first iteration. But then managing expectations for later problematic because things later take longer (e.g. Customization). Clients don't understand this.

### **LULLABOT THEMING TRAINING – April 18 2010**

- Get DVD that compliments this course; also check out [www.lullabot.com/drupalbox/theming](http://www.lullabot.com/drupalbox/theming) (presentation slides are here)
- Knowing Unix command-line conventions really helpful to theming in Drupal – a fast way to find and move files.
- Lots of university participation at training course.
- Know shortcuts to clearing the cache. The devel module is key here.
- Custom theme placement: sites > all > themes > mytheme
- Note that this file structure is required for Drupal multi-site functionality to work. The 'all' is the key here – need this file structure for multi-site to work.
- Never, ever, ever touch Drupal core. Never touch the core modules or core themes folders.
- .info file super important. This is a simple text file. Tells Drupal that your theme is there and what files your theme uses.
- Drupal uses a 'clear-block' hack. It is the clear-fix class – it is exactly the clear-fix class in proper CSS. Clear-fix class is called clear-block class in Drupal 6. It is named properly as clear-fix in Drupal 7.
- Be aware of what you name your region variables because you could blow away default variables that you shouldn't.
- Recommend that always start with a layout that includes the following regions: left, right, content, header and footer. Highly, highly recommend starting with these defaults. Note cannot get rid of content region – this is a default in Drupal and is required by Drupal.
- In terms of features in the .info file, you can turn these off and hard code relevant items (e.g. Logo, tagline, etc.) so that folks can't change these. Features are not very practical!!!
- Recommend creating specific 'css' and 'js' directories instead of relying on the Drupal default directory. Makes it easier to locate and maintain css and js files.
- Do NOT put any conditional stuff into the .info file. Do NOT do this.
- Modules come with CSS that you may or may not want; use Firebug to determine what CSS is generating the presentation. The prefix name of the CSS files should match that of the module name.
- The .info file lets you control overriding what CSS is displayed. Allows one to enable – or not – some CSS.
- Having too much CSS causes performance issues. There is lots of CSS (e.g. Installed modules cause more CSS to become part of your Drupal implementation). Recommend to aggregate CSS to optimize performance. Note that aggregating can cause problems with the design process – solution is to turn on the aggregating at only the production stage. Also note that IE doesn't support more than 29 CSS files; there is a module that handles this for IE.
- About .tpl.php files. This is to alter the HTML. Remember these files are for HTML!!!
- There are several .tpl.php files in Drupal. For example, block files, node files, page files.
- In general, you really want to AVOID having multiple page.tpl.php files. Try to have only one page.tpl.php file. The only exception may be if you have a really, really different front page – where you may need to create a page-front.tpl.php
- Modules can and should provide .tpl.php files. A good module will have a .tpl.php file (rather than having hard coded HTML as part of the module). The .tpl.php file allows you to modify the HTML for the module (e.g. If a module comes with table code and you really need the layout to be handled via CSS)

- Naming convention for .tpl.php files ... proper way is 'node-nodetype.tpl.php'; so, more concretely, 'node-blog.tpl.php'
- Note that the Drupal default is node.tpl.php – need to use the following convention for additional customized node types: node-nodetype.tpl.php
- Name your content types very carefully. This refers to the machine names for content types. You want to make sure that the names for content types match the names for the corresponding .tpl.php files. This helps you identify what goes with what within Drupal.
- Note to be careful in Drupal about where you use hyphens and where you use dashes in naming conventions.
- Highly, highly recommend using CCK.
- About the template.php ... Drupal standard is NOT to close PHP. This is IMPORTANT. Closing the PHP will cause forms-handling functionality to break. As well, you can have a little bit (the key here is only a little bit) of logic or basic SQL queries (e.g. # of users online) ... BUT in general use the proper locations for extensive logic and complex SQL queries. Extensive or complex SQL queries should be handled by modules.
- Do not theme to id numbers generated by Drupal. Causes problems when moving sites from server to server ... if other sites exist that relied on the id numbers generated by Drupal ... all sorts of stuff will get overwritten. e.g. ALWAYS AVOID a name like page-node-1.tpl.php ... instead, give the file a name such as page-node-edit.tpl.php
- Create a region. Go to .info file. Add region here. Name region what you want – make sure it makes sense to site maintainers. Definitely use underscore for the region name – required because PHP variable. Use PHP underscore convention. Variables in PHP are case sensitive – use lower case. Now go to blocks administration area and you will see that the regions exist. Next you got to go to your page.tpl.php file and add <?php print \$bottom\_left; ?> etc. Use Firebug to see how things end up placed on the page. Remember to put blocks in region.
- Remember to design for tabs; Drupal puts in tabs everywhere.
- Recommend starting with an existing well-done theme. The Zen theme is a common starting point for sites.
- Preprocess functions: the template.php file. A changing site slogan – example shown in course. The template.php goes in same folder as .info file. Pretty much purely PHP – working with PHP functions. This is using a preprocess function – the particular type of function you're working with here. This function mythemename\_preprocess\_page(&\$variables) is recognized by Drupal because of the naming convention. There is a variables array within Drupal that is referenced. Remember to empty cache – or won't see changes applied.
- Example of how to change what someone sees based on who they are – their login credentials.
- Must worry about security when creating a theme. This is a key point of vulnerability for Drupal sites. Mainly because designers tend not to have worried about security historically. As a Drupal themer, got to know about security and vulnerabilities that your theming could make your site open to.
- Drupal is all about creating patterns for managing your sites. Make up, define and document the patterns for creating and maintaining your sites.
- Recommend the use of pathauto for the creation of nice SEO friendly URLs.
- Create the HTML and CSS separately, make sure it works, then make it into a theme in Drupal. That is, make a fully functional HTML template along with the CSS (e.g. In Dreamweaver) – and then turn this into a Drupal theme.
- Be familiar with api.drupal.org
- Really got to know PHP to do stuff with your theme. You can do basic stuff without, but in general really do need to know PHP to do the things you want to do – to make content/presentation appear on your site how you want.
- You don't have to know all the preprocess function stuff. There is a bunch of stuff and ways to do things. Some things could be done as content types instead of PHP. Remember, there is lots of stuff you can do with just HTML and CSS and content types in Drupal. Though two of the presenters say need to know PHP or else. One presenter of the 3 says remember can do a lot to make great sites without knowing PHP – get as far as you can, and learn the PHP when you get to a point when there is something that you need to do that requires PHP.
- Two modes for nodes: teaser and full.
- Turn off the theme developer tool. Breaks pretty much all of the js in Drupal. Go to modules page and turn it off. Breaks file uploading functionality too – the theme developer tool. But really need the tool to create the theme!
- Breadcrumb example. Don't want double arrow. theme\_breadcrumb function. See function information at drupal.org. Select text from drupal.org and copy it into template.php. Do not accidentally grab PHP close character – will break your template.php file. Rename the function so that Drupal knows which to use. mytheme\_breadcrumb is the naming convention you want to follow. Puts hearts instead of double arrows for the breadcrumb path.
- Summary: start w/ HTML and CSS files; then create page.tpl.php file; used node.thp.php file; then template.php for variables – this is where the logic is – using preprocess functions; finally looked at theme functions (e.g. Breadcrumb function)
- Start with markup you want, then swap in the variables you need.
- The main process is copy the original and then modify for your theme.
- Must haves:
  - Firebug
  - Devel module
  - Theme devloper module
  - CCK

#### Course feedback:

- moderate to good – they know their stuff but run into having trouble explaining some things; nice that they are so excited about Drupal and how 'cool' it is – but sometimes get caught up in coolness and move quickly missing steps in explanations
- don't show how to do exercises – just say go to it, do this (would be helpful for them to walk through example first and then have us apply, or to have manual/handout)
- there is no handout or manual with the exercises to follow; no way for participants to follow steps at their own pace; and no way to look back and see what steps one may have missed
- a couple of modules were needed for the course; no warning in advance that these should have been installed for the course; spent time getting modules installed during course because needed in order to do exercises... then needed to catch up on what was going on in course ... think missed some steps ... and no manual to check; really seemed like they forgot that we would have to install these modules to do the course exercises (b/c they work with the modules all of the time and forgot that need to be downloaded from drupal.org site)
- morning break was very close to lunch – way too long without a break
- no paper notepads or pens provided with conference – assume that all attending will be attending with laptops (is this really the case?)

#### Questions and answers:

- `template_files` – want own dynamic template feature. Course really focused on adding to Drupal to achieve functionality you want. Example of how to create node-teaser variable. Copy `node.tpl.php` file and rename as `node-teaser.tpl.php`. `<marquee>` Said not great example – would likely do different way to achieve different display of teaser. Example of how to make moving/rolling teaser (i.e. Ticker or ticker-tape teaser)
- what css framework is the best. Fusion is great for skinr module. Some like 960 grid. 5 or 6 popular ways to go. Got to figure out what will for you. No real answer to this. Others love Blue Print. Bacis theme is really great. Zen is good b/c it has a lot of commenting – this is really helpful.
- The zen approach. Fantastic starter theme. Particularly if new to Drupal. All the comments could be bothersome if you already know lots about Drupal theming.
- Didn't talk about base and child theme. How to make a base theme. Should know how to do this. Oranges and apples themes. Define as base theme in `.info` file. Create base theme for university. Then department needs own theme – do not make a whole bunch of themes. Instead use a parent theme and have children themes. Use this with multi-site installation. Have only one base theme for university/company. Talked about Grammy base theme. The basetheme reference in the `.info` file is where you define which is the base theme – doesn't need to be inside the other theme to have parent-child relationship (this changed with the `.info` file in Drupal 6). Can also have grandparent, parent and child – all sorts of possibilities.

#### Common mistakes:

- Don't make things more complicated than needed. Make sure to use css where possible. Don't go nuts with `tpl` files. Really avoid making `tpl` files. And don't make too many regions – just what you really need (they're still being rendered even if not being printed – really does not optimize performance having too many regions).
- Crazy design with too many layouts. Crazy or too many page layouts. Can use block configuration to manage. Chose which pages blocks show up on what page. Use this method. If get really complicated, may want to start looking at panels (panels module).
- Remember to clear the cache. Or you won't see your theming changes.
- Use `pathauto` for URLs. Associate with themes. The flowers section, versus food section, versus wine section. Can get really far with this. No need to go with `tpl` files.

**Getting Started with DrupalCon, Monday, April 19, 2010 Presenters: Joshua Brauer, Brenda Boggs, Roberts Douglass, Heather James, Kenny Silanskas, Michael Haag, Peter Bull, Kent Gale**

- Large session; required overflow room
- Background: started by Dries Buytaert (original creator and project lead for Drupal); open source; many sites using Drupal (Best Buy, Amnesty International...); thousands of modules, which are customizable; and it's a community (emphasis on community!)
- Drupal jargon/terms – node, views, themes, region, blocks, menus, taxonomy, core, modules, permissions, hooks, cron, Druplicon (mascot)
- Brief slide outline how Drupal works at high level along with its components
- Custom content – every site is special!
  - CCK – build on core; through interface or get fancy with APIs; fields can be different fields
  - Taxonomy example:
    - Flavours
      - Chocolate
        - Dark
        - Milk
      - Vanilla
- Reusable content: robust, saves time, think of it as **recycle** your workload, **reuse** your content, and **reduce** time; stored in database. Showed examples of sites: film festival, and kitten
- Drupal's social skills: like a Miss Manners Guide; good at sharing, good communicator, understanding, good listener, makes friends easily.
- Drupal and e-commerce: sell products, event registration, etc.
- Search engine optimization: website only useful if you can find it, Drupal core and contributed modules help to do this; make sure no duplicate content on site URL Redirect
- Multiple languages: 63% users other than English; interface to be translated such as blocks and views; 100% duplication is possible
- Acquia: Drupal Gardens; install Acquia Drupal, which is free; AMP (Apache, MySQL, PHP) or DAMP (for Mac and Windows); [acquia.com/downloads](http://acquia.com/downloads)
- Hosting providers mentioned
- Community and Resources: its strength, 600 groups, IRC, #drupal, documentation, handbooks, businesses; [drupal.org](http://drupal.org)
- Drupal 6 is current; Drupal 7's new features include usability enhancements, CCK being part of core, and more.

**Panel: Many flavours of Drupal Training, Monday, April 19, 2010 Presenters: Michael Anello (DrupalEasy), Alex Urevick-Ackelsberg (ZipTec), Jeff Robbins (Lullabot), Ryan Price (DrupalEasy), Greg Knaddison (GPS), Tom Geller (Lynda.com)**

- Providing training for difference audience; one size doesn't fit all
- Years ago power users were php coders but now more diverse backgrounds; lots of books to get started
- Different types of training represented on panel: public training, private training, in-house training, client training – brief discussion of each
- Training existing staff for Drupal is possible (ie. java > php)
- Follow-up done by all panellists by survey; suggested web form module for survey; realize can't keep everyone happy (too fast, too slow – hey, we understand in IST-CS); good to do pre-assessment beforehand – perhaps 4 or 5 questions; ask for 'quotes' from satisfied clients
- One panellist said it is possible to train anyone to instruct – (really?)

**Case Studies in Social Commerce: Waste Management's Greenopolis & Mattel.com, Monday, April 19, 2010 Presenters: Rohit Anabheri (WM's G), Jeff Fulton (M)**

**Greenopolis** – believe in giving back to the community – raise consumer awareness and action; wanted increased social media for engagement (YouTube, FB, Twitter)

- **Rethink** (social media – use CCK media’s video upload module), **recycle** (kiosks; work in development to have mobile apps such as an iPhone apps to scan products before curbside), **reward** (Ubercart) by giving points on account, coupons and gift certificates (for hotels, dining, car rentals, or to donate check)
- **Why use Drupal?** – flexible (modules for everything), popular (open source, large community, other Waste Mgmt teams considering the switch), and powerful
- **How used?** Modules – TweetMeMe, Service links, Facebook Connect modules; Zen-based themes and sections module
- Went from v5.15 to 6.x now: had over 100 modules; used DRUSH and bash scripts, 2-hour downtime; rebranded at same time.
- 3 years now using Drupal. No longer asking Why using Drupal? but **Why not use Drupal?**

**Mattel:** wanted to go from corporate to interactive; 6 million visitors/month; checked Joomla!, Alfresco, Drupal; chose Drupal and relaunched the site; partnered with a digital agency called ‘Work at Play’.

- **Strengths** of Drupal: SOLR search integration; fast powerful caching features; easy to edit/upload/customize; thriving community; shared libraries; easy to script CMS controls; most imp > node queue functionality key to CMS success
- **Challenges:** corporate perception of Open Source; difficult to create a work flow for Dev > Stage > Production; no downtime
- **Now:** mattel.com not only corporate speak but an aggregator; shared branding; faster, video portals (ie. commercials), games/toy factory; Barbie I Can Be – Dressup

### **State of Drupal, Monday, April 19, 2010 Keynote presenter: Dries Buytaert**

- What’s changed in the last 2 years (since Boston’s Drupalcon)? RDF (Resource Description Framework) in Drupal 7 for a semantic web and linked data; more companies like IBM and Accenture getting involved (Microsoft announced: a new SQL server driver for PHP with beta PDO; Acquia Drupal on its MS Web Platform); book ‘Pro Drupal Development’ will be updated to Drupal 7
- In March 2010 – how many Drupal sites? Installed crawler to count Drupal sites and found 1 million so that means Drupal powers 1% of the web; current success is due to Drupal 6.
- Drupal 7 – many years of work, accepted 6611 patches from 732 different people; 25 contributors for 50% of patches (displayed slide of names and asked those on the list to stand); ‘a culture of passion and believing’; honourable mention to Angela ‘webchick’ Byron; more than 70 contributed modules moved to core, which means lots more code and that means slower but more scalable. As of today, 114 critical issues left. When released? Best case scenario is June 2010 while worst case is Fall 2011. Drupal 7 will drive adoption!
- % of sites: WordPress major chunk, then Joomla! then Drupal.
- Cloud Computing – Passionate
- Have to be on top of the game to stay relevant; mentioned a book “Innovator’s Dilemma – when new technologies cause great firms to fail” by Clayton Christensen; provides a framework, some point the product over satisfies the consumer and lower-end product often causes the original product to fail (eg. Mainframes > PCs); it’s Enterprise over Drupal > Drupal over WordPress; choice? Go upmarket? Follow market (challenging)? Or do both? Or stick with the low end market.
- Next? To succeed in enterprise, need to focus on missing features; for low end market need to focus on better experiences; need to grow, learn and deal with ?? but adds complexity
- There’s a culture of ...sharing and collaboration, passion and believing, self-organizing, of innovation, of execution (talk is silver, code is gold!), and a culture of having fun. “Awesome happens!”
- Next DrupalCon is Copenhagen, Denmark August 23-27
- Very inspiring and energetic talk.

### **Drupal 7 – What you need to know, Monday, April 19, 2010 Presenter: Angie ‘webchick’ Byron**

She started in 2005 as a student, is D7 core maintainer, Drupal Association board member, co-author of ‘Using Drupal’, works for Lullabot.

**Agenda:** FAQs, Drupal 7 demo, core developer, how to help

- Timeline: Feb ‘08 D6 released and D7 opened; Sept ‘09 feature freeze (code thaw), Oct 15 ‘09 code freeze (API cleanup), Nov 15 ‘09 polish phase. Now for the release date – still 214 critical issues need to be handled first (in alpha3).

- Can I start alpha testing? Yes. Should I start porting modules and themes? Yes. If it's #D7CS, then this means there's a module pledge on day D7 released.
- What about upgrades? No upgrade path yet from D6>D7 although being worked on. Upgrade shouldn't break data.
- Demo installation of Drupal 7, which went quite smoothly.
- Mentioned accessibility – not perfect but much better
- Encourage testing.

## Case Studies in Publishing: Time Out NY and Taunton Press, Monday, April 19, 2010

### Time Out NY

- PowerPoint presentation footer: 'We're hiring'; major competitor is New York Magazine.
- Reason for choosing Drupal? Usual reason (community, open source...); 90% of those who chose product now gone.
- Legacy: Adobe suite, and then home-grown CMS. Note: used cut and paste from Adobe suite > home-grown CMS
- DB – 4D (French product that is being used by London office); and using K4 cross-media publishing platform.
- For print > using Drupal; [Treehouse Agency](#) introduced the feeds module for easier editorial tools; SOLR for faceted search
- Haven't launched yet. Lost ¾ of CMS team.

### Taunton

- Publishing company, primarily print but also digital
- Used Ektron (CMS .net) previously
- Why Drupal: looked at challenges, leaders, noted PHP #4 language (after C, Java, C++)
- As of Oct 2009 – still migrating
- Using Acquia; allows them to put resources into other areas by using modules
- Phase 1 > pilot > review and improve > phase 2
- Challenge: moving .net to Drupal; migrating without downtime
- Key > training; having advocates of CMS is a good idea; minimize core customization; use methodologies and best practices; partner to mitigate risks (they used Acquia and Drupal Professionals)

## Planning and Executing a Successful Drupal implementation, Monday, April 19, 2010 Presenters: Michael Morris and David Leonard, Phase 2 Technology

- The hard stuff: big, complex Drupal projects
- How to work with stakeholders
- Executing good project processes on difficult Drupal projects
- Defining scope and requirements
- Addressing common situations

Have worked with 80 plus Drupal projects

- Give proper intro: terminology; explain open source; community; explain 'free'; modules don't solve all problems; Drupal isn't magic; it's a tool and platform; good practices
- Setting expectations properly!!! Ramp people up; build up functionality; polishing; getting content ready; timelines set appropriately
- Communicate effectively – have a project portal; status reports; regular meetings
- Drupal can be difficult to grasp at times; views – not easy; spend time; module developers don't always understand what you want
- Train stakeholders: admin guides; too much documentation can get stale; starter guide – quick, like the ones that come with cameras; screen casts great but take time – ie. create small snippets of some task
- Encourage early stakeholder testing: all details are correct; taxonomy may need some flushing out; no sub for someone using it; design should look good and converted into a theme correctly
- Need stakeholders involvement; make them part of the team; help them help themselves; hold them accountable
- Understandable the Drupal learning curve; may hit a wall; prepare for mistakes – they happen but you will learn
- Put the right team together – not just coders; soft skills necessary; multi-disciplined team that understands the Drupal ecosystem
  - Project mgr

- Analyst - solutions architect; understanding the problem for a workable solutions; defining content types; step a bit into dev mode
- Designer- not always a full time
- Web producer
- Developers – get them segmented
- Dev team – good if possible
- Don't be a baseball team – encourage role flexibility; more like soccer team
- Be agile and manage scope effectively – not a Drupal only thing; work iterations – in chunks; firm on scope but flexible on features
- Foster intra-team communication; real time communication
- Insist on quality – is everything covered
- Start digging – what are the site's goals, who are the audience, focus of content...
- Requirements toolbox – site map, wireframes, specs, system diagrams, design; create a site map to define the breadth of a site
- For layout – brainstorm and sketch layout ideas – white boarding
- Create detailed wireframes – best way to define site; designers may have a different take
- Don't neglect details – more time spent upfront will save time later
- Document integrations with system diagrams – what APIs to use and how to talk with one another
- Document the specifications – formal such as a requirements doc;
- Meet design expectations – if not a lot of feedback from wireframes so may need to visualize with colours

#### Addressing common challenges

- replace bad habits with good ones
- Structure content properly
- Permissions
- Use taxonomy effectively - web server ... automatic tagging but certain vocabularies may not jive
- present a clean admin interface – help text is important; free of jargon
- give them a good editing experience – get a realistic preview; don't show all menu items – small toolbar so not to overwhelm
  - Allow editors to curate and innovate
  - Provide for editorial workflows
  - Represent content relationships correctly
  - Taxonomy
  - Differentiate between authors and users, as necessary
  - Improve search with Apache SOLR – for faceted search; can exclude; can prioritize
  - Pay attention to roles and permission – esp with a large site
  - Scale back all permissions at once and then grant gradually as comfort level increased
  - Migrate content accurately!

#### **Show, Don't Tell – Guerrilla Usability Testing, Tuesday, April 20, 2010 Presenter: Zoey Kroll**

- Started as a consultant for Deloitte, quit her job, and went to the woods and lived in a tent for 3 months; came back and started using Drupal
- Usability testing is a 'blast'; totally jazzed about
- No longer believes in tests because they feel right or wrong; websites are conversations
- Part of DrupalCon SF committee; [Hayes Valley Farm](#) project; Edible Office project

The current state of a project – get a snapshot of current site; use Google Analytics

Research and testing methods

What does it all mean?

Quick site revamp

Share the story- hope to tell stories; interact with your users (who are people!)

Talk to each other

User Experience of Hayes Valley Farm – used to be a land with a freeway thru it; mayor wanted gardens; donations of trees, etc. Have volunteers

A case study: website and physical site; rapidly evolving; dynamic community

---

- Research and testing – did an online survey but not a good response; then questionnaire with people working – 15 minutes so people had to stop working to do so not a good idea; asked for words that came to mind to help with branding (neighbourhood, urban, green...); then used the camera and did 30 second interviews with 1-4 people helping, what are their goals and vision;
- User testing is platform independent;
- Showed video – DrupalCon attendees went on a tour; were impressed and inspired; like the Drupal community – ‘inspiration is like a fire’; interviews are the story; wanted a simplified home page so it resulted in big change; connecting to your community so built into the process – it’s not about the ‘module’; make changes to the site; project plan went out the window because they were making changes so quickly.
- Summary – go talk to the people – don’t be afraid to get out from behind the desk;
- Q&A – how do you know the people are the website users? It was a question asked when interviewing.
- What type of questions do you ask? What are you passionate about? Why are you here? What is Hayes Valley Farm to you? How do you see yourself plugging into the project? What do you see happening in a month?
- Earth Day is Thursday so party at the farm
- Did people need to sign a release form? Yes, the standard form
- Audience member said she likes the story telling
- There’s other usability testing methods but they chose to get out from the desk
- What’s next? Will continue doing interviews, will continue changing site; don’t need permission to change the site; could ask previous interviewed persons if they thought the website was better
- 30 sec interview videos was a lot better than long paragraphs; ‘community run farm’ is the message

We then had to interview someone near you. I met David Alcala, Adobe (on the Dreamweaver CS5 team)

### **Case Studies in Academia: Drupal at ASU & John Hopkins Knowledge for Health, Tuesday, April 20, 2010**

**Presenters: Jeff Beeman, Guy Chalk and others**

- Drupal in Academia – 4 people; large audience from universities
- Jeff spoke first; story of ASU (Arizona State University – 100,000 students); unique story and lots of success on campus
- Web consulting services within main IT dept at ASU; they do shared web hosting; lots of drupal sites; contract work too; hosts their sites; provide core services; wrote and maintain a module (ASU Webauth + Drupal) and ASU (Zen) Theme; all on campus use same header and footer – well most use it; enterprise Drupal environment – HR warehouse; PeopleSoft; do support and consulting – one co-employed with Pres Office; support users but also contracted support with depts.; also share with Drupal community; try hard to keep community strong
- Have Drupal centre web site for support.
- Background: asu.edu in 2006 – had proprietary; decentralized IT; all different looks; recognized problem with users; new CTO came and had a strong vision – need consistent nav and look
- 2010 – now a little variation; consistent nav and header and branding
- Biggest obstacle: politics! Territorial site ownership, no incentive to unify, no supported tools, no community
- The approach – make it as attractive as possible. The velvet glove aka carrot vs stick. Can’t dictate. Make drupal so tasty people can’t resist moving to it.
- Did so through a combination: tactic #1 – tools (stnd platform of Drupal and modules); stnd environment (shared hosting);
- Tactic #2 – support community; web community started; ASU Drupal users group (large membership of 20-50 at each meeting – 100 members); Drupal office hours; training with Lullabot/ASU; support and outreach (went out of their way to engage users and help with problems – no magic formula – everyone embraced – they were lucky)
- Got buy-in from Library, Pres Office, News site; key players
- Web hosting – 25 servers with approx 100 sites each – 2500 domains; high performance servers (direct admin servers) with only one site on the server
- Core sites in drupal:libraries, news, student services
- Enterprise environment
- Moving into the cloud - Amazon
- Consistent look; global templates; Drupal sites use central them
- Only support Drupal and none others

- Consistency in apps (modules)
- Repurposing of text
- Future: need to clean up mess made and have 500 sites not always maintained; growing pains due to number of sites; cloud hosted services; change is slow and difficult; institutional perception (Drupal is open source who have negative opinion); prove Drupal as legitimate; need to focus support efforts (need to deliver better support resources), training; mobile is a big deal and challenge trying to solve
- Positive outlook – feel good where they are at; although scary at times
- Q&A – dashboard? Just getting into it now.
- office hours? How long to ramp up? – reaction to people asking for help, and then having a dedicated time. Not a good model anymore because now more support in other areas to be done. Have a ticketing system, and some email; trying to expand the scope of online resources including FAQs, IRC channel (marginally successful for hardcore people),
- infrastructure? How sites deployed? Most account owners have installed Drupal where they provide packages with WP, media wiki, Drupal, modules Silo environment.
- How many programmers? His team is 5 people; graphic designer, 2 developers, - lots of Drupal developers across campus.
- Do you have restricted data? For student data or financial data they need to go thru enterprise environment They have an architecture review board.
- User group? Does it relate to a larger phoenix group? People wanted to come during work. The Phoenix group is that large.
- How to manage inexperienced Drupal web users? Yes, and there's problems with dept setting up own site and not realizing the scope. They now tell users that content providers may need to know CSS and basic html. Looking into more.

Johns Hopkins U – Guy Chalk, TJ Moyer, Larry Lee Michael Haggerty...

- Drupal for grants area
- Person speaking from Comm Programs and Knowledge Mgmt
- Based in Baltimore with 31 field offices worldwide; 60 active projects (malaria, HIV/AIDS...)
- Each project wants a website but can't support with the amt of funds from grants Can't create a large web team due to cost; different budgets and different funding agencies
- Grant up to 5 years; have databases to continue;
- Have disconnected silos of information; low budget static sites; etc
- Therefore moved to Drupal due to open source, quick start up, large dev community
- Sites within sites – Org groups, toolkits, domain access; interoperability, SOAP and Web services
- Using taxonomy, auto-tagging, and community IDs
- Google search appliance collections; custom search engines
- Over 1-1/2 years wrote modules – InMagic Import, File Cart, OG Toolkits, KSS Search, E-learning
- Michael Haggerty – legacy data import module
  - Docs going back to the 70s that needed to be indexed in Drupal; and how to connect to these systems (to be accessible); need to do the work when less traffic
  - Created InMagic – runs cron; manual import feature allows them to select a db and start an import of all records that have been created or modified in a legacy system by date or by specific id number; update existing nodes – refetching them;
  - File cart module – not using Ubercart due to the overhead cost; checks country and then allows access to certain docs
- TJ Moyer – OG Toolkits (organic groups – OG) contains contributed modules; allow new toolkits to be developed and deployed quickly; example- Haiti Relief Toolkit was developed over a weekend and then updated with feedback
- Larry Lee – KSS Modules is a series of Drupal modules for a single search engine for the silos. Started to integrate with Google Search Appliance; currently supports several Google products; discussed why going with an external search engine
- Q&A – how long to launch? Some smaller sites already so developers got up to speed quickly; took a year to define sites; took a little over a year to launch

**BOF – Drupal at Universities and Colleges, Tuesday, April 20, Chair: Scott E Worthington**

- Large turnout but room too small (and hot)

- Spoke with Oregon State University; previously used DW templates; now use Drupal 6, and use WordPress Mu for blogging [oregonstate.edu/cws/webtrain/presentations1](http://oregonstate.edu/cws/webtrain/presentations1)

**Shhh! This is a Drupal-powered Library Site: Wednesday, April 21, 2010. Presenters: Tammy Allgood at ASU, Katherine Lynch at Drexel Univ, Amy from Huntsville-Madison Public Library, Alabama**

- Chose Drupal 3 years ago
- 7 libraries at 4 diff campuses
- Used ColdFusion – 29,584 files; and DW templates
- New system requirements: easy way to update, low costs, sep of internal content from external content, wysiwyg, wanted no additional software, workflow process, low development and maintenance
- Learning Drupal – fairly steep learning curve, readme.txt was first source of help incl the books too, good user group that brings in training ie Lullabot.
  - Created great local community and used this community
- Migrating content – weed! ; got people involved with enthusiasm, had tax already mapped out; Testing – made sure content was there
- Taxonomy – setup a subject taxonomy; roles sample shown; Access Control Module used
- Lots of documentation although not everyone reads, style guide
- Publishing workflows – easy to setup in 20 minutes; people seem laid back – just publish it; workflow module and trigger module
- set up Drupal environment (determined modules, created theme, determined content types, views, taxonomies)
- listed modules – 13 or so; likes Captcha and works fine for visually impaired (reads it)
- how many content providers? About 40
- training? About an hour. Use WebEx due to various locations

Amy – Huntsville-Madison Public library IT dept

[hmcpl.org/tech](http://hmcpl.org/tech) – notes and questions

- no budget, no people, only her

highlights --Rick Astley would never give you up, let you down, run around and desert you... so she kept stats as to success with graphical results displayed so when dollars turned could be used; 2 weeks employees had to take off due to budget problems so had to make it easy for those who had to do the input easily; Ubercart used for donations, memorials, and ticket sales

Katherine Lynch at Drexel Univ Libraries in Philadelphia

- group study areas, etc background info and slides
- e-dbs – 551.
- user testing – usability testing with 1-to-1 testing – tried quick and dirty 3-second tests with paper prototypes; also in-depth interviews for 30-40minutes and got their background info such as user level, web use habits, favourite sites, most visited sites
- need to respect your users and know their needs – to deliver resources to patrons
- for the catalogue – intuitive searching, faceted browsing, social connections
- support staff – patrons wanted answers quickly;
- library usage is different now – want to know hours, phone numbers, faqs, calendar of events, group study room reservations
- personalized library experience – looks interesting

**Harnessing Drupal for your next web project , Wednesday, April 21, 2010. Panel ist: Drew Gorton, Michael Caccavano, Bill Fitzgerald, Eric Gundersen**

- Bill – FunnyMonkey; community sites focus
- Drew – Gorton Studios; due the complete project
- Michael – Treehouse Agency; big media sites
- Ian/Eric – DevelopmentSEED; consulting

Drew- uses basic setup, focus on how to engage community; 100 websites;

Bill – made 30-40 sites so far; takes 3 months – 15 months depending scope of project;

Michael – takes several months to a year; 15 sites so far but huge sites

Ian – made 60-80 sites; takes month – 9 months

What helps the most when the client approaches you? Goals, website focus, budget/RFPs, how the organization is staffed (building process, maintenance for ongoing); decision maker (who is the key person for single pt of contact); what content do you want; requirements – how big or small is this; static/dynamic; wireframes

How to gain trust and charging? Trust is the bldg block; discovery process is important and is chargeable; then use experience to see what the client can do and what he can do for you; scoping a project may take a couple of days but depends on the project size; develop a partnership; works both way.

What do I do if I am the Drupal expert? Do these principles work also? Yes, then non-tech people can understand. Can get the support needed.

Communication with client? Ticketing system is the best for data trail. Comment – Basecamp not recommended.

Where/how does documentation happens? Use notebook (wiki) for clients, details for modules, etc and then copy/paste for client; document the code; must need to document the scope during the discovery process; constant refinement;

How many people work for you? Mentioned numbers. Less than 15 per business.

Should you use Drupal 7? The Examiner is building on Drupal 7; depends on features and what modules available; maybe start on Drupal 7 and then add other features later; depends on project and timing; launch time.

### **Drupal in Government by Jeff Walpole and Chuck**

Does Drupal work with government... yes, sort of

Who is using? Recover, IT Dashboard, etc in US. Big in Europe, esp Belgium due to Dries. Lots in New Zealand. Data.gov.uk.

Regulation drives a lot of what we can and cannot do – not technology driven.

Government is under more scrutiny and pressure on the web than ever before. What is and isn't acceptable for the government to use.

Special government considerations – security (huge), privacy, FOIA, records mgmt, accessibility (section 508), accuracy, various regulations.

Reason to go to Drupal – tech control and flexibility, sharing/OSS community innovation, cost savings, meet user demands/expectations, avoid vendor lock in, speed of development

Now look at what government needs – **meet citizen expectations**, tech control and flexibility, sharing/OSS, cost effective solutions, no vendor lock in, speed of development

An open government (OG) changes everything! Contrasts what techies view of what OG is about, such as data, transparency, etc.

Open data plus open source doesn't mean open government.

Tech requirements of OGD (open govt initiative) [www.agency.gov/open](http://www.agency.gov/open), such as best practices, use modern technology, publish open government plan, downloadable machine readable docs, etc.

3 ways to publish open data: publish raw data, create an online catalog, and make data human and machine readable.

OG workshops: enables the public have 2-way conversation with gov't; being proactive in publishing to the web; collect needs/ideas of the citizens; improve citizen services; being open with info/data/policy decision making.

/Open Concept Site – make agencies compliant on time, template approach to not missing key elements of the OGD, make it easy to update, change, publish, follow RSS; collaborate

The wcms is just part of the entire tech stack- have legacy systems, enterprise architecture, etc – although Drupal does lots of it.

Issues in implementing Drupal in Government – by Chuck (works for Acquia)

- Open government is cool. Drupal is cool.
- Organizational issues: fear and uncertainty around open source; many existing relationships already (contracts, technologies); contracting and program mgmt; cultural shifts (have other skills); not a monolith (different requirements depending on Pentagon, congress, white house, etc); security is a big deal (July 4<sup>th</sup> cyber attack; after state of the union address attack); technical issues (scale, security, section 508, privacy – ie can't use GA due to can't use cookies, politics, records management.)

How Drupal is affected by government use – improved reputation as a professional tool, growth of user base and size of community, exposure to a much larger set of less tech users...

### **Open Source in Government keynote, Wednesday, April 21, 2010 Speakers: David Cole, whitehouse.gov and Andrew Hoppin, NY State Senate CIO**

- Mentioned how we all can affect change
- Dave Cole, whitehouse.gov project – champion for Drupal in government
- Working with Drupal in 2007; wanted to work with the movement; audio of Obama talking about Google founders and how ordinary people can do extraordinary things; change comes from the bottom; saw how people got together during the campaign to discuss ideas
- Site features:
  - Blogs – groups
  - Briefing room - using apache solr,
  - Special features
  - Disclosures
  - Live video – see what's happening; used to be backroom conversations
  - Public feedback
  - Open for questions – stats on page show number of questions, etc
  - Public voting
  - Web forms – Captcha
- Modules powering the website – all noted on screen
- Everyone contributed! Had people stand who contributed the modules they use.
- Next step – contributing code now. Giving back. Modules (four)- content http headers; akamai (cache related), govdelivery (replaces stnd Drupal mail send function with web service calls); node embed (allows for content layering) . Available at Drupal.org.
- Why Drupal – scalability, core base, security.

Dave Poll, Mike Walsh and Andrew - panellists

Discussing open source adoption in government; can partner with government (federal and state levels)

### **Drupal 7: UX Wednesday, April 21 2010 Presenter: Dries Buytaert (original speaker not available)**

- Lab testing done (university?)
- Usability isn't the best before 7
- Inability to find content after reading it
- Mental model of creating pages not expected
- Users expected wysiwyg
- A usability team is born approx 15 people to make D7 better
- Drupal team is good for incremental updates but needed to do something more – a holistic approach
- Hired Mark Boulton and Lisa xxx – more than 6 months looking at usability
- D7.org – user experience project web page
- Goals tackled – make most frequent tasks easy and less frequent tasks achievable; design for the 80%, privilege the content creator, make the default setting smart

- Content managers – not site builders but end users (writers, editors, etc); Designers and developer
- D7 with simplified startup options (standard or minimum); password verification made more clear (better site credentials); info architecture improvements; recently added content made easier to find (now one click away with the content tab); point and click edit; add shortcuts to most used locations (like bookmarks and is configurable); overlays (admin layer and site layer) and have a visual distinction – can turnoff the module if prefer; sitewide dashboard is customizable; clearer content/help menu; reorganized forms – vertical tabs keep it clean; primary actions separated from tabs.
- D7 designer and developer changes – adding features (modules) workflow improved – direct links to permission and conf settings; clearing cache more accessible (more prominent than D6 – no need to scroll); upgrade manager – can do so without leaving Drupal
- Drupal gardens – quick and easy way to get started, wysiwyg, media support (can be uploaded,reused..)
- Drupal 8 – respect the IA

Happy with D7 and it's an improvement from D6.

### **BOF – Training in Higher Education, Wednesday, April 21**

- Another large turnout but room too small; time crunch
- Everyone submitted an index card with name, email addr, area of focus; two people evaluated focus areas and then we broke into smaller groups based on focus area
- Nancy from Stanford U, myself, and Luke from a small N Carolina College (name?) discussed training.
- Stanford had trouble with getting people to attend training, and then the websites not done correctly; no enthusiasm for websites; no mandate. Discussed some solutions. They highly recommended Lullabot training and DVDs.
- Luke was the only person for their college CMS so not too many issues although he has no backup person.
- Lynda.com > training good for core group, developers
- Training for Drupal developers: problem with multiple ways to do things. Standardization is necessary! Perhaps accredited training for developers
- Tools for remote learning; Screenshare, GoToMeeting...
- Solicit feedback to measure effectiveness
- Documentation, cheat sheets, creative commons, use the advanced help module

## Highlights (by categories)

### Drupal project planning

There isn't a right way of breaking down a Drupal project for everyone.

#### *General break down of a Drupal project*

- **Users:** accounts, permissions, roles
- **Nodes:** Entities of content (typed, structured content)
- **Lists:** List of entities matching certain criteria (filtered collections, blocks, events, etc)
- **Actions:** Actions users can take on entities (vote, forward, buy, etc)
- **Structure:** Structure needed to organize things properly (navigation hierarchy, menus, regions, etc) Drupal doesn't care about structure; this is why pre-planning is so important:
  - o Defining regions of the site (Wireframes)
  - o Defining roles
  - o Defining rules for flow of navigation and functionality
  - o Defining Content Types and Taxonomy
  - o Define navigation structure

#### *Make up of a Drupal team*

- **Architect:** Plan the big picture (defines which path we are going to take to a set of goals)
- **Builder:** Configuration and setup
- **Developer:** Code the custom bits (usually overlaps with Builder role)
- **Designer:** Turn ideas into a visual design and then into digital media (Photoshop, html/css)
- **Themer:** Turn output from designer (Photoshop files or html/css files) into a Drupal theme. Translates the static design vision into a Drupal template.

## Drupal development/customization

Whether you are developing a custom module for a site, creating a theme, or developing a contributed module, knowing the APIs that are at your disposal will save you a lot of time and improve the quality of your work.

Drupal core modules as well as many contributed modules allow re-usability of quality code by implementing an API interface.

When writing code implement an API interface so other developer are able to re-use your code. Become an API designer.

## Security

Cross Site Scripting vulnerability (XSS) and Cross Site Request Forgery (CSRF) are the more common vulnerabilities in Drupal implementations.

### *How to protect a Drupal site*

Themers:

- Read tpl.php and default implementations
- Rely on your module developer: the developer is responsible of passing variable values already filtered

Developers:

- Know where the text come from: DB, URL, web service
- Know who can change it: user permissions

- Know the context it's being used at: Mail, DB, Web and server context. Proper way of handling strings: [http://crackingdrupal.com/sites/crackingdrupal.com/files/filtering\\_text\\_0.pdf](http://crackingdrupal.com/sites/crackingdrupal.com/files/filtering_text_0.pdf) (more info: <http://acko.net/blog/safe-string-theory-for-the-web>)
- Don't delete, archive and provide undo
- Truly destructive actions should be hard to do
- Use db\_rewrite\_sql to fix SQL injection problems.

## Theming

Themes can be implemented as:

- Functions: faster
- Templates: easier to work with. Recommended when starting working with Drupal.

Important: Don't use PHP includes in template files.

Converting Drupal 6 themes to Drupal 7: <http://drupal.org/update/theme/6/7>

## System administration

There is no way of having a zero-touch administration setup.

Tools available for running production servers minimizing access to them:

### *Drupal tools:*

Tools for managing a Drupal system without login to the production server:

- Features
- Spaces
- Strongarm
- Drush
- CTools

### *System tools:*

Tools for managing large deployments. Managing many servers consistently at the same time without login to each of them:

- Configuration management: Puppet (configuration on the .htaccess file is moved to Apache config file so that the configuration can be consistently managed from the Apache configuration)
- Remote execution tools: Func (Fedora Unified Network Controller)
- Log management: Centralize the logs using Syslog. Use Splunk to generate reports, search, etc the logs.

### *Development tools:*

- Version management:
  - o Git distributed version control system
  - o Atlassian Bamboo
- Build tools
  - o Phing
  - o Built-in PHP tasks: PHPLint, PDOExec, etc.
- Continuous integration: making sure the build tools are working all the time
  - o Syntax Checking
  - o Testing
    - Simpletest
    - Selenium
  - o Packaging
  - o Deployment
- Use Drupal Developer installation profile for getting a development setup up and running fast.

Limitations of these tools:

- Features have to be enable
- DB updates: DB rollbacks must of the time require human intervention

### Drupal distributions

Drupal distributions allow for easy Drupal installations: download a single file (containing core, contributed modules and themes), unpack it, visit install.php, done.

Drupal distribution system uses Installation profiles, Drush and a packaging system.

Drupaldistrowatch.com keeps ratings and reviews of Drupal Distributions and Installation profiles.

Example of Drupal distributions:

- Acquia Drupal
- Open Atrium
- OpenScholar: Please watch the video at <http://scholar.harvard.edu/> This may help us with the Faculty member personal websites.

### Search engine

Two options for replacing core content search and getting better performance and extra features including faceted search:

- Apache Solr: runs outside Drupal environment which avoids overloading of drupal server if ran in a different physical server.
- Search Lucene API: easier to install but integrates tightly with Drupal, so, there is no way of passing the load to another server if needed.

### Sessions

1. Getting started with Drupal(con) by Joshua Brauer (Acquia), Brenda Boggs (Acquia), Heather James (Acquia), Kenny Silanskas (Acquia), Michael Haag (Acquia), Peter Bull (Acquia), Collin Waid (Acquia), Chris Rutter (Acquia), Robert Douglass (Acquia), Jeffrey McGuire (Acquia)
2. How Drupal Works: An Architect's Overview by Jeff Eaton (Lullabot)
3. 20 APIs Every Drupal Developer Should Know by John Snow (Trellon)
4. AJAX and Javascript in Drupal7 (for developers) by Randy Fay (Randy Fay, LLC), Rob Loach (Acquia), Katherine Bailey (Work At Play)
5. Drupal site security for coders and themers by Greg Knaddison (Growing Venture Solutions), Peter Wolanin (Acquia), Ben Jeavons (Growing Venture Solutions)
6. Default theme implementations: a guide for module developers who want sweet love from Morten and JohnAlbin by John Albin Wilkins (Palantir.net)
7. "Don't Touch that Server": A toolkit for zero-touch production environments by Chuck D'Antonio (Acquia)
8. Leveraging the Chaos tool suite for module development by Roger López (Rent the Runway)
9. Views for developers by Larry Garfield (Palantir.net)
10. Apache Solr Search Mastery by James McKinney (Evolving Web), Robert Douglass (Acquia), Peter Wolanin (Acquia)
11. Build a Powerful Site Search with the User-Friendly, Easy-to-Install Search Lucene API Module Suite by Chris Pliakas (CommonPlaces e-Solutions, LLC)
12. Token for Fun and Profit in Drupal 7 by Jeff Eaton (Lullabot)
13. Drupal Distributions on drupal.org by Derek Wright (3281d Consulting), Kevin Reynen (Open Media Foundation), Chad Phillips (Apartment Lines), Jeff Miccolis (Development Seed)
14. Drush by Dmitri Gaskin (Boldium, LLC)
15. iPhone, Drupal, and Web Services by Kyle Browning (Workhabit)
16. Best practices in contrib development and support by Dave Reid, Randy Fay (Randy Fay, LLC), Greg Knaddison (Growing Venture Solutions), Derek Wright (3281d Consulting), Jennifer Hodgdon (Poplar ProductivityWare)
17. Module Building for Beginners (Drupal 6 and 7) - Intro and Workshop by Chris Shattuck (Build a Module.com / Implied By Design LLC.), Ezra Barnett Gildesgame (Growing Venture Solutions)

## **Workshop**

The second part of the session "Module Building for Beginners (Drupal 6 and 7) - Intro and Workshop" was a Hooks Demo which showed how to develop a module using hooks for adding or extending functionality. The demo used a website where users were able to submit food recipes. The goal of the demo was to redirect users to the appropriate category (taxonomy term) page after submitting a recipe. Although this wasn't a hands-on workshop it was very useful for understanding how the hook mechanism works in Drupal.