1:1 Subject Matter Expert

Ned Holtes

Ned has been the lead developer on a number of projects which involved responding to the customer request for a CMS solution. In one specific case he worked directly with the product design lead as well as end-users of an international company. During this engagement he showed the capacity to understand their needs and provided them with a solution which involved a content management system.

Due to his direct relationship with end-users and the success of the project, we felt he would be able to articulate his perceived needs of CMS administrators and retrospect on how his work addressed those needs.

Learning Goals

Throughout the conversation, we will be listening out for (and probing when appropriate) to learn:
1. What are a CMS Admin's basic needs?
2. Where might they run into issues when using a CMS?
3. How can the CMS design help data governance?

Take-Aways

From a technical perspective, Ned shared the following considerations as important
1. Early efforts into uncovering the *actual* needs (as opposed to the articulated/perceived needs) can save significant time, money and effort
   a. Asking questions like "what should be edited", "how frequently does it change" "what does it need to look like" and "how fast do the edits need to be live" help expose whether a CMS is necessary at all--there are other (faster, cheaper) ways to solution the need based on the answers to those questions
   b. Understanding the need for specific visual design needs vs many stylistic options (bullets, colors, etc) is important to everything (the end-user, the admin, and the developer)
2. Many needs are at root very basic: Add, edit, delete quickly.

Open-ended Discovery Interview Script

Thanks, general icebreaker discussion, then remind that today the focus is the interviewee's experience in CMS development and specifically how the Administration user's needs informed their approach.

When thinking of the role, how would you describe a "minimum viable CMS content manager" (capabilities, mindset, etc)?
- Anyone can do this job, the work is in discerning what they really need
- General needs: add & edit content within a predefined framework
- Keep them focused - what are they actually asking for ("In many cases, people would really like an admin tool to edit things, not necessarily to make a brochure website")

What are some of the key customer needs you consider when building for those who will manage CMS content?
- In his experience it's up to the developer to "Get specific with people--they tend to want the world *but* not need that much in reality"
- Speed of the software
- Question their assumptions: "Does it really belong in a cms in the first place?"
- Do they want to make stylistic changes to things (like bold, bullets, etc)

How would you describe the most common customer pains you try to address when building CMS?
● Frequent updates
● Wanted it to look *exactly* one way that the CMS doesn't afford

What would you describe as "quick wins" when building the first rev of a CMS?
● Delivered very quickly, as little styling as possible
● Show how changes can appear immediately
● Lock down the styles and *then* spend some cycles improving the few visual styles available for consistency & polish
● Considering how the CMS integrates with other products services or aspects of the site

What sort of information do you look for when considering what fields and functionality should exist in a CMS?
● "What do you want to edit?"
● "How frequently do you want to edit it?"
● "How quickly do you need to see it show up?" (maybe you don't need dynamic after all)
● Developer comes back with an informed approach based on their understanding of CMS capabilities, sometimes CMS is not even necessary

How do you find CMS architecture related to data governance and reporting goals?
● Requirements need to be considered as first-tier, these can be more important than the data model
● Most people do not have those requirements at all, most OTS CMS tools do a terrible job with that stuff
● If admin change, tracking, etc is important, it's not really a CMS

What did I not consider?
● "Customer requests for 'CMS' are generally a trap--really pin people down hard on what they want and *then* figure out if the solution is a CMS"
● "Spec must be cemented...reliant on the content model...changing this information in the future is really challenging"
● Early consensus & expectation setting is important
  ○ Late breaking requirements can completely derail the process which results in an "architectural abomination" or huge churn
  ○ Articulate what the user is not able to do, ex: "you can not arbitrarily re-order things in the catalog"