KM COLUMN

FEBRUARY 2009

Custom code, CMS and portals

In the early days, intranets and websites consisted of pages and pages of static content. Originally published by hand, many of these sites migrated to a content management system (CMS) in due course.

Over time, more sites started to include a range of rich interactive functionality. This included publishing content from a database, providing online 'calculators', or incorporating interactive Flash-based elements.

The immediate challenge was working out how best to create and manage these sites. Many sites ended up being entirely custom-developed, including both the front end and back end.

Then products such as portals came along, offering a single platform through which all functionality could be delivered. These were particularly popular on intranets, promising to simplify the delivery of interactive features.

Content management systems also became richer in their capabilities, providing much more than page publishing out of the box. They too offered a development platform for custom development.

Organisations are now confronted with many different ways of delivering custom code and interactive features on a site. Each have their strengths and weaknesses.

This article outlines three major approaches, and summarises the characteristics of each.

In practice, these can be mixed, and individual circumstances will dictate which approaches fit best. The goal of this article is therefore to provide a guide for decision making, and to encourage further research.

James Robertson is the managing director of Step Two Designs, an intranet and content management consultancy based in Sydney, Australia. James specialises in intranet strategy, web content management, information architecture and usability.

There are many ways of delivering custom code on a site

The growth of interactivity

All kinds of websites are increasing their use of interactive features and functionality. This includes:

- information dynamically published out of back-end databases, such as stock prices or data tables
- e-commerce systems, booking engines or other online applications
- map-based displays or other interactive elements

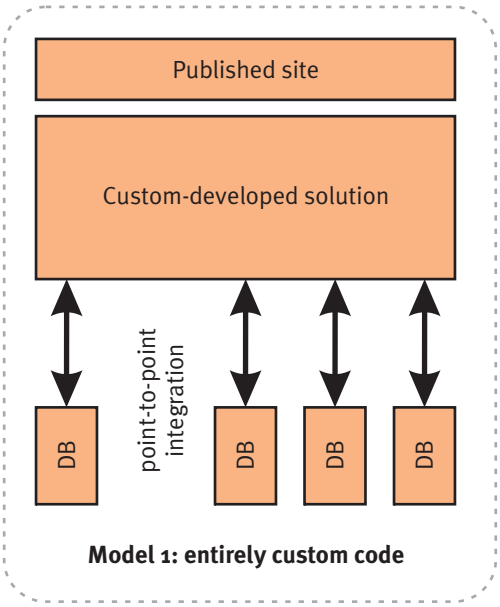
In many cases, intranets are richer sites, delivering a broad mix of business tools and online processes. These commonly include:

- self-service access to business systems such as HR, payroll or finance
- room or resource booking
- staff or expertise directories
- collaborative tools, including team spaces and wikis
- front-line business applications and CRM systems
- real-time information displays driven by back-end systems
- personalised dashboards

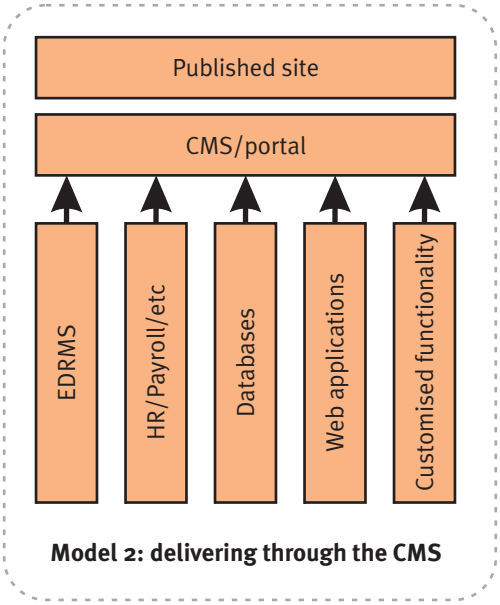
These features, by their design, often require custom code to be developed. This may be needed to connect to specific back-end databases, or to deliver functionality in a specific way.

Other functionality is delivered by third-party applications, such as HR or finance systems. These need to be tied into the sites, ideally to deliver a seamless user experience.

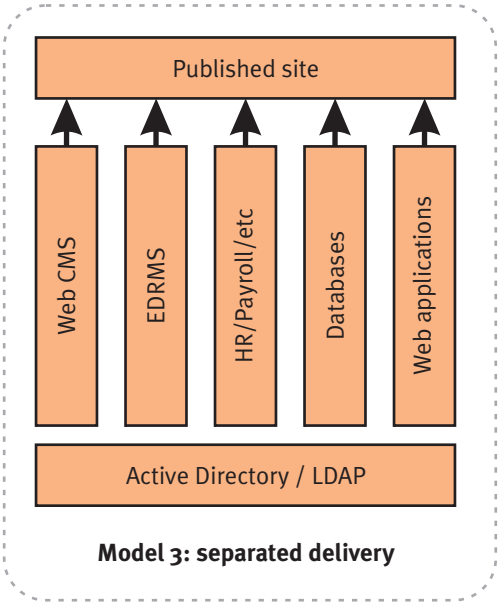
In both cases, questions arise on how to manage the development or integration, and what role the CMS or portal should play.



Model 1: entirely custom code



Model 2: delivering through the CMS



Model 3: separated delivery

Three approaches

Looking across many organisations, three main approaches to managing custom code or interactive elements have been used:

1. *Entirely custom code*, where both the front and back ends of the sites are custom developed.
2. *Delivering through the CMS or portal*, where a single product acts as a platform or gateway for all functionality.
3. *Separated delivery*, where separate systems sit alongside each other, connecting at the front end of the site.

The diagram on this page summarises each of these approaches, and they are explored in greater detail in the following sections.

1. Entirely custom code

In this approach, the entire site is developed, including both the front end (published site), and back end (authoring and management).

Where necessary, integration is developed with external systems or legacy solutions, as part of the overall solution.

Note that in many cases, developers will reuse existing code or solutions, reducing the amount of entirely new work required.

A range of developer libraries and underlying frameworks will also be used to reduce development effort.

Advantages

This approach offers a number of important advantages:

- Perfect fit for needs

  Custom developing a solution allows specific needs to be met, without the constraints of third-party products or platforms.

- Ideal for addressing unique requirements

  Where needs are unique or uncommon, development may be the only way of meeting them.

- Complete control over platform

  Developers can often find CMS or portal products constraining or difficult to use. By custom developing a complete solution, developers have complete control over every aspect of the solution.

- Very tight integration

  All elements of the site work together, as they have been developed together. This eliminates the need for difficult integration between separate products.
- May allow richer solution to be delivered

  When done well, a custom solution can provide a richer solution than an out-of-the-box CMS or portal, particularly for interactive or dynamic functionality.

Development is inherently risky, in both cost and time

Disadvantages

There are also a number of disadvantages:

- Increased development risk

  Writing code is inherently risky. Bugs, technical problems, cost and time overruns are common. Taking a completely custom route greatly increases the amount of development required.
- Longer project duration

  Without the ability to exploit out-of-the-box capabilities, custom-developed solutions can often require greater time and effort.
- Common functionality has to be replicated

  While custom code is valuable when addressing unique or specific needs, it must also be devoted to replicating common capabilities. Typical back-end functions, such as authoring, link management, versioning and workflow must all be built, potentially a considerable 'reinventing of the wheel'.
- CMS products are complex

  There are a lot of details and features to deliver to replicate the functionality of CMS or portal products. Many of these small points can have a significant impact on user satisfaction or product stability.
- All costs paid by organisation

  All development costs must be covered by the organisation that commissioned the work. In comparison, the development underpinning a commercial solution is shared between hundreds of clients.
- No upgrades or bug fixes

  Additional development costs must be paid to address bug fixes or to add additional functionality. There is no commercial provider to regularly release updates.
- Solution is stationary

  Custom developed solutions are stationary: they don't move unless additional money is spent. Meanwhile, the broader industry continues to innovate, eventually outstripping the custom solution.

When to use

Custom-developed solutions are ideal for uncommon or unique problems, where an out-of-the-box product doesn't fit requirements. In these cases, a developed solution can deliver considerable innovation and business benefits.

Many products aim to provide a single gateway for the user

2. Delivering through the CMS or portal

In this approach, the CMS or portal is used as the platform for all development. This may involve:

- using the product's API to build specific functionality
- integrating directly with the CMS or portal to deliver functionality to the end user
- customising the product itself to deliver required functionality
- using built-in product functionality to eliminate the need for some custom development

In all cases, the CMS or portal acts as a the single 'gateway' to the site, with all functionality delivered through the product.

This was promoted heavily as the great advantage of portal products, which promised to greatly cut down on development costs and integration effort.

There are also many content management systems that are specifically targeted to this kind of development, emphasising their support for customisation and integration.

In most cases, the goal of these solutions is to minimise back end work while delivering a consistent user experience for site visitors.

Advantages

There are a number of advantages of this approach:

- Reuse of in-built functionality

  The CMS provides a range of useful 'core' functionality, such as security, workflow, or content publishing. If exposed through an appropriate API, this can help to simplify and streamline creation of new functionality.

- Tight integration with the site

  Delivering everything through the CMS or portal makes it easy to deliver a seamless site from the end-user's perspective, with the same design and navigation throughout.

- May reduce the number of products

  Working with a single CMS or portal can help to cut down the profusion of individually coded systems that can grow over time.

- May simplify development

  Depending on the specific product, developing within the CMS or portal framework may reduce development efforts (although this is often more of a promise than reality, see below).

Beware of vendor lock-in, in a rapidly changing marketplace

Disadvantages

There are also a number of significant disadvantages to this approach:

- Site functionality is tightly coupled to the CMS or portal

  By developing within the CMS or portal, site functionality becomes tightly tied to the product itself. If there is a move to replace the CMS, this forces the redevelopment of all custom functionality, as there is no option of migrating this to the new system or leaving it in place.

- Over-investment in the CMS

  The result can be an over-investment in the technology platform. Because of the changing marketplace, all products have a finite life, and it may not be good value to invest in the product long-term. Since the customised functionality is inseparable from the product, the value of this is also reduced.

- Greater development complexity

  As outlined above, the CMS can provide core functionality to speed up web development. In practice, however, it would appear that programmers are commonly confronted with a steeper learning curve when developing within these solutions.

- More development required

  With the CMS managing all functionality delivered to the site, everything needs to be integrated with the CMS. This can greatly increase the amount of development required.

- Constrained development environment

  Developers must work within the bounds of the CMS or portal, rather than in a purpose-created development environment. This may limit the tools or approaches than can be used.

- Proprietary development less valuable for career path

  A considerable body of knowledge needs to be established to work within a CMS, regardless of the product used. With so many products in the marketplace, these skills may not be usable in future jobs or in other organisations. This makes it less attractive for developers, which may make it harder to obtain staff.

Don't over-invest in a single technology platform

When to use

Use this approach when the portal or CMS offers the combination of powerful out-of-the-box capabilities and an attractive development environment.

Fundamentally, the additional effort of working within the product must be outweighed by the immediate benefits delivered.

Summary of approaches

1. Entirely custom code	2. Delivering through the CMS	3. Separated delivery
Advantages: • perfect fit for needs • ideal for addressing unique requirements • complete control over platform • very tight integration • may allow richer solutions to be delivered	Advantages: • reuse of in-built functionality • tight integration with the site • may reduce the number of products • may simplify development	Advantages: • separates CMS and developed code • reduces vendor lock-in • simplifies development environment and process • provides greater freedom for code development • reduces the learning curve for developers • recognises that integration is uncommon
Disadvantages: • increased development risk • longer project duration • common functionality has to be replicated • CMS products are complex • all costs paid for by organisation • no upgrades or bug fixes • solutions is stationary	Disadvantages: • site functionality is tightly coupled with the CMS or portal • over-investment in the CMS • greater development complexity • more development required • constrained development environment • proprietary development less valuable for career path	Disadvantages: • limits ability to benefit from CMS capabilities • makes it harder to provide a seamless user experience • may increase integration effort
When to use: When addressing unique or unusual requirements which cannot be met by out-of-the-box solutions.	When to use: When the benefits of out-of-the-box product capabilities outweigh vendor lock-in and greater development complexity.	When to use: Increasingly the recommended approach for sites looking to benefit from out-of-the-box product capabilities and additional development efforts.

Be cautious of pursuing this approach on the basis of platform consolidation, as this often doesn't eventuate in practice. Using a single product also delivers limited benefits in terms of integration or providing a 'single source of truth'.

3. Separated delivery

In this approach, efforts are made to keep the content management system or portal separate from any custom-developed code.

The CMS is used to deliver pages of content, and other out-of-the-box functionality. Additional development is done on separate servers, with a range of techniques used to combine all the elements on the published sites.

In this way, a consistent appearance is presented to users on the published sites, while delivering this behind-the-scenes from a number of sources.

By tying together the elements with a common source of user information (such as Active Directory or LDAP), a measure of 'single sign-on' is achieved.

Advantages

There are a number of advantages to this approach:

- Separates the CMS and developed code

  In this approach, there is a clean line between the CMS and the developed code, allowing the two to evolve independently. For example, the CMS can be replaced without forcing a redevelopment of all web applications.

- Reduces vendor lock-in

  With less commitment to the product, organisations have greater flexibility in modifying or replacing the underlying technology platform.
- Simplifies the development environment and process

  Web applications can now be developed in an environment specifically set up for this task. All typical development methodologies and tools can be applied in a standard way, without having to consider the specific considerations of any given CMS.
- Provides greater freedom for code development

  Decisions about hosting, development platform or language for web applications can now be made fairly independently of the CMS. As long as there is a mechanism to deliver the code to the site via one of the ways outlined later in this article, the code development process can be highly flexible.
- Reduces the learning curve for developers

  Developers no longer have to build an in-depth knowledge of the CMS, allowing standard development practices to be applied. This reduces the learning curve for developers, and potentially increases the attractiveness of the work for prospective and current staff.
- Recognises that integration is uncommon

  Although many projects have a high level focus on 'integration', direct integration between products only rarely necessary More important is the appearance of integration on the site itself, which can be achieved in a variety of ways.
- Reduces project bottlenecks

  By allowing custom code to be developed independently of the CMS, project activities can be run in parallel. This eliminates many dependencies, and mitigates project risks.

The appearance of integration is often a primary goal

Disadvantages

There are a number of disadvantages to this approach:

- Limits ability to benefit from CMS capabilities

  By developing applications completely separately from the CMS or portal, the product's features cannot be leveraged. This may require some additional development to replicate these capabilities.
- Makes it harder to provide a seamless user experience

  Unless managed carefully, developing solutions separately can lead to a lack of consistency and integration on the published sites.
- May increase integration effort

  Where integration is required, it will need to be custom developed, instead of making use of out-of-the-box capabilities.

Create the appearance of integration

When to use

This is the recommended approach in many cases. Vendors themselves are increasingly guiding their clients in this direction, recognising that custom work within their products limits their product evolution as much as it impacts on their clients.

As a general rule, look for opportunities to decouple CMS products and custom applications. This reduces long-term risks and simplifies decisions relating to application development.

When using this approach, however, a strategy must be put in place to ensure that site users are still provided with a seamless experience. See the following section for a number of approaches.

Integrating applications with the site

The key challenge in taking the third approach is integrating the applications or interactive functionality with the site. There are a number of different mechanisms that can be used as appropriate:

- Linking to applications

  The simplest approach is to link to separate web applications. If these use the same page layouts and CSS, the user is unlikely to notice any significant change. Only appropriate for entirely free-standing applications.

- 'Skinning' applications

  The CMS can be used to 'wrap' or 'skin' separate applications with the standard design of the site. This can generate a very seamless appearance, but has some technical wrinkles.

- Embedding code in published pages

  Many CMS products now support the ability to drop 'chunks' of code into published pages, either as part of the body of the page or in the page template. This can be the most flexible way of incorporating dynamic features into the site.

- Using the product's API

  In the most complex situations, the API of the CMS or portal can be used to integrate directly with the product. If care is taken to do this at 'arm's length', then the impact of CMS changes can be minimised (if not entirely eliminated).

In practice, a mix of these approaches will be needed, depending on the specific task at hand.

Conclusion

This has been a simple overview of a complex, often technical, topic. The goal has been to outline the broad approaches that can be taken, and to highlight common strengths and weaknesses.

When considering what approach to take in practice, identify both the long- and short-term objectives of the site. Use these to find a path that will take the greatest advantage of the tools available, while avoiding long-term lock-in and increased development complexity.

Wherever possible, take an approach that gives flexibility in swapping out individual elements, and be careful of any 'silver bullet' claims from technology platform advocates.