A Modular Microservice Architecture for Multi-everything Content Management

Sixth International Conference on Creative Content Technologies
May 25 - 29, 2014 - Venice, Italy

Dr. Hans-Werner Sehring, T-Systems Multimedia Solutions, 28 May 2014
Outline.

- Modern Content Management Requirements.
  - Content Management Today.
  - Extended Requirements of Modern Applications.
- Microservices.
- Concept-oriented Content Management (CCM).
- CCM Microservices for Multi-everything Content Management.
- Summary and Outlook.
Deutsche Telekom. Partner for connected life and work.

Deutsche Telekom delivers one-stop services and solutions: for all customer communications needs – at home, on the move and at work.

T-Mobile

T-Mobile offers cell-phone solutions in the Netherlands, Austria, the Czech Republic, and the USA.

Telekom

The Telekom subsidiaries provide products and services for the fixed network, mobile communications, the Internet and IPTV in Europe.

T-Systems

T-Systems delivers ICT solutions for major corporations and public-sector organizations worldwide.
T-Systems Multimedia Solutions.

- 1250 employees.
- 9 offices.
- €122 mio. revenues.
- 18 years of experience.

*Internet agency ranking of Germany's Bundesverband Digitale Wirtschaft (BVDW).*
MODERN CONTENT MANAGEMENT REQUIREMENTS.
Modern Content Management Requirements.

- **CMS today**: foundation of web sites.
  - media-agnostic content management.
  - processes, internal and across system boundaries.
  - editing, tools, and workflows, rights and roles.
  - publication, rendering, and playout.
  - systems integration.
- **Extended requirements** of modern applications.
  - marketing.
  - sales.
  - support, knowledge management.
  - extranet: closed user groups, SSO, ...
  - intranet: SSO, self service, ...
Content Management Today.
Content Management Today.
Technical View on Content.

- Initial idea of content management: in documents separate
  - content,
  - structure, and
  - layout

1 Introduction
1.1 Def. of Content
Content is something miraculous.
Let’s structure it.
Further aspects of content modeling are relevant for web-based content dissemination:

- **Navigation** (for primary navigation, breadcrumbs, sitemap, ...).
- **Variants** of content.
  - language variants.
  - full and short versions for full view, teaser, mobile site, ...
- **Multimedia documents**, eventually in variants (e.g., print, online and preview).
- Editorial **parameters**, e.g. for directed search (“searchandizing”).

For documents, mobile apps, mobile sites, etc. similar aspects have to be taken into consideration.
Extended Requirements of Modern Applications.
There have been four inflection points marking the history of the corporate website:

1. Informational
2. Transactional
3. Persuasive
4. Contextual

[ektron, *From Content to Context to Engagement*, Whitepaper]
Extended Requirements of Modern Applications. Domain Requirements.

- **Targeting, marketing automation:**
  - Adaption of presentation in order to **make content accessible more easily:** content representation, page layout, search results, ...
  - Optimization of content selection and presentation in order to make system fun to use by offering **optimal user experience**.
  - **Leading the users** to specific content (and finally to **conversion**).

- **Multi-site management:**
  - Landing pages, microsites, campaigns.
  - **Reusing** content, software, and web components, **optimize operations**.

- **Cross-channel marketing:**
  - Shifting focus from **Point of Sale (PoS)** to **Point(s) of Contact** in order to attract users.
  - Iteratively **measuring** success and **improving** a site w.r.t. specific goals.
  - Learn about **user journeys**.
Extended Requirements of Modern Applications.
Software System Requirements.

- Technical answer to the domain requirements: **multi-everything**.
  - Multimedia.
  - Multilingualism.
  - Multi-site / Multi-brand Systems: Multi-tenancy.
  - Multi-device.
  - Multi-channel, Cross-channel.
  - Personalization, Targeting.
Extended Requirements of Modern Applications. Multimedia.

- Studied well in the document world.
- Multimedia models, languages, etc.
- Sometimes disassembled into monomedia artifacts in CMSs to account for:
  - **different sources for artifacts**: video from streaming service, time codes from CMS, comments from social media, ...
  - **multi-language** (e.g., different audio tracks for movie)
  - **multi-device** (e.g., different video encodings of movie)
  - **personalization** (e.g., subtitles to video)
Extended Requirements of Modern Applications. Multilingualism.

- "Same" content in different language, locales, etc.
- **Typical pattern:**
  - master copy in default language
  - localized variants referring to it
  - with localized variant fall back to default language
  - not only content; also, e.g., layout: RTL languages etc.
- **Open:** translation process
  - actuality
  - localization of prices, availability, legal constraints, ...
  - => multiple departments involved
- **Open:** variants of multimedia artifacts (image, video, ...)
  - are they localized?
  - all artifacts, or only some?
Extended Requirements of Modern Applications. Multilingualism Example.
Extended Requirements of Modern Applications. Multi-site / Multi-brand Systems.

- For different reasons there are content management platforms hosting more than one site:
  - multi-site: the main web site, a corporate web site, microsites, ...
  - multi-brand: present a family of products
Extended Requirements of Modern Applications. Multi-tenancy.

- Technically, Multi-site / Multi-brand Systems are forms of **multi-tenancy**. These tasks are accomplished by multi-tenant content management systems.
- Different from multi-tenancy as in, e.g., databases, where applications are isolated from each other.
- **Options:**
  - Shared content (e.g., products) / protected content (e.g., financial figures).
  - Shared functionality (e.g., search) / specific functionality (e.g., order forms).
  - Shared data (e.g., global search) / specific data (e.g., site-specific search).
  - Central editorial board / distributed editors.
Extended Requirements of Modern Applications. Multi-device.

- Web sites are offered in a form that does not only ensure correct display on the various stationary browsers, but also on mobile devices.
- Above warranting the correct display some sites demand for device-optimized mobile variants, e.g.:
  - using the device OS’s native look and feel
  - using specific hardware features
- Alternatives:
  - native Apps
  - responsive design
Extended Requirements of Modern Applications. Multi-channel.

- Same content is published on different media.
- Pull publication: web, mobile, API, SEO information, tracking, ...
- Push publication: social media, newsletter, ...
- Next step: **cross-channel**, have connected channels, track **user journeys**.
Extended Requirements of Modern Applications. Personalization.

- The goal of targeting / marketing automation is primarily reached by means of personalization with the goal of contextualizing content presentation.
- Determine which content and in what presentation is most appealing (helpful...) the user in the current usage context.

"Content is king" ➔ "Context is king"

**Context shortens sales cycles, produces higher quality leads, and grows revenue. Context increases customer engagement and loyalty.**

A “content management system” is no longer enough to successfully attract and retain your audience. Managing content will no longer be enough.

**Content + context will drive successful engagement**

[ektron, From Content to Context to Engagement, Whitepaper]
Extended Requirements of Modern Applications.

- In field of content management: **Web Content Management (WCM)** now named
  - **Web Experience Management (WEM)**
  - **Customer Experience Management (CEM, CXM)**
- Currently of particular interest: personalization, marketing automation, or similar.

Semi-automatic product approaches:
- **Marketing Automation (Adobe)**
- **Create/Deliver/Optimize (eZ Systems)**
- **Adaptive Personalization (CoreMedia)**
In short, the microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. These services are built around business capabilities and independently deployable by fully automated deployment machinery. There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies.

[http://martinfowler.com/articles/microservices.html]
Microservices.
Common Characteristics.

- Not a new approach, can be seen in many places. Currently consolidation of experiences and identifying common characteristics.

- Goals:
  - **no SW monoliths** in the sense of statically linked software modules (e.g., layered architecture), but components with independent lifecycles
  - service-oriented approach, structure **software (and teams) according to business capabilities**, not functionalities
  - support “agile architecture”, **design for evolution**

- Uniform API (“published interface”), required for orthogonal composition of system building blocks.

- P&F style communication with “smart filters, dumb pipes”: encapsulated logic, simple protocols.

- No shared knowledge; e.g., local storage instead of centralized databases => no common database schema, no commonly agreed-upon conceptual models.
CONCEPT-ORIENTED CONTENT MANAGEMENT.
Concept-oriented Content Management (CCM). Requirements for Personalizable Systems.

- **Requirements:**
  - Users want to adapt (*personalize*) content to their current working context individually, while
  - existing content is preserved and
  - cooperation between users is maintained.

- **Demand:** *open* and *dynamic* environments
  - **Openness:** models ...
    - are not limited to predefined concepts and
    - can be changed at any time.
  - **Dynamics:** content management systems ...
    - follow model change without interrupting the domain experts’ work and
    - maintain existing contents and communication structures.
Concept-oriented Content Management (CCM). Orthogonal Personalization.

- **Content personalization:**
  - content, concepts, schema, context.
  - includes presentation personalization.

- **Structure personalization:**
  - Schema (attributes, relationships, constraints).
  - Categories (create new, recategorize).

- Implemented through openness and dynamics.
Concept-oriented Content Management (CCM). Contributions.

- The key requirements of openness and dynamics are met by the **Concept-oriented Content Management (CCM)** approach by means of its key contributions:
  - **Modeling language**
    - Modeling performed by domain expert.
    - Open for changes.
  - **Model-driven system development**
    - Incremental generation.
    - Fully automatic, without developer intervention.
  - **Software architecture**
    - Component-based.
    - Evolution friendly.
Concept-oriented Content Management (CCM). Logical CCM Architecture: Components

- CCMSs consist of **components**
  - one component for each domain model
  - cooperation for domain combinations
  - integration to model revisions and to derive variants

![Diagram showing component structure]
Concept-oriented Content Management (CCM). CCMS Implementation: Modules

- **Modules** implement components
- Reconfigurability at runtime for dynamics through:
  - *Separation of Concerns* by module kinds with distinguished functionality,
  - uniform module API, and
  - statelessness of modules.
Module kinds for the construction of CCMSs:

- **Client module**: Mapping to the generic concepts of a *base system*
- **Server module**: Provision of functionality according to a standardized interface
- **Coordination module**: Delegation of requests according to a certain strategy
- **Adaptation module**: Conversion between different models
- **Distribution module**: Connection to remove module of a (distributed) component
Concept-oriented Content Management (CCM). A Microservices View

- CCMSs follows a **microservices approach**.
- It exhibits the agreed-upon **properties of microservices**:
  - Components encapsulating business capabilities
  - Architecture made for evolution
  - Separation of Concerns in implementations (modules)
  - Uniform published interface and decentralized internal schemas
    <-> CCMSs’ component (server module) and internal module interfaces
- CCM’s **additions**:
  - Caring about components’ microarchitecture in order to allow *dynamic* evolution.
  - Making remote access explicit through distribution and server modules.
CCMS Architecture as a Microservices. Modules as Filters.

- **Module interface**: API generated from the content model.
CCMS Architecture as a Microservices. Modules Combinations as Filter Chains.

- **Module combinations**: The generated API with pre-defined functionality allows to combine modules freely. They are arranged in a layered architecture, where the access pattern resembles pipes and filters (pull chain).

![Diagram](image)

- Module 1 uses Module 2
- Module 2 uses Module 3
- Module 3 uses Module 4 and/or Module 5
- Content flow

(pull) request to module 1

Response of module 1 (content)
CCMS Architecture as a Microservices. Distribution Modules as Pipes for Remote Calls.

- **Distribution**: explicit through module combinations incorporating distribution modules. Other module combinations use implementation language bindings (direct calls).

(module 1 uses module 2)

(module 1 uses module 2 remotely)
CCMS Architecture as a Microservices. Uniform component API.

- **Component API**: Component APIs are defined by server modules according to some standard interface, e.g., as a RESTful service.
CCM MICROSERVICES FOR MULTI-EVERYTHING CONTENT MANAGEMENT.
CCM Microservices for Multi-everything Content Management.

- The demands for
  - Multimedia
  - Multilingualism
  - Multi-tenancy: Multi-site / Multi-brand Systems
  - Multi-device
  - Multi-channel, Cross-channel
  - Personalization, Targeting

can be addressed by a CCMS with the usual means.
CCM Microservices for Multi-everything Content Management.

Multimedia.

- Media can be stored in different asset management repositories, remote locations, ...
- A coordination module combines content into aggregated assets.
- If the content structures differ, adaptation modules unify the structures. Example: pictures without timing information are related to sound (with timeline).
- Then, multimedia presentations can be created. Ex.: Timing can be added to the pictures.
CCM Microservices for Multi-everything Content Management. Multilingualism.

- Content for different languages is hosted in different repositories, accessible through client modules.
- Adaptation modules change the schema, so that language variants of content refer to one “master content” in the default language.
- A coordination module selects content from one of the base modules according to a chosen language. Language variants are related to the “master content”.

<table>
<thead>
<tr>
<th>server module: multi-language</th>
</tr>
</thead>
<tbody>
<tr>
<td>coordination module: language selection</td>
</tr>
<tr>
<td>adaptation module: en</td>
</tr>
<tr>
<td>client module: en</td>
</tr>
</tbody>
</table>
CCM Microservices for Multi-everything Content Management. Multi-tenancy.

- Content is accessed by one or more client modules.
- Content is delivered by one server module per tenant.
- One coordination module (or multiple) provides the corresponding view on the content.
- Eventually one adaptation module per tenant establishes the schema required by its tenant.
CCM Microservices for Multi-everything Content Management. Multi-device, Multi-channel.

- Content is delivered by one server module per device/channel.
- One coordination module per device/channel augments the main content with device-/channel-specific variants (e.g., shorter/longer texts, smaller/larger images, navigation, ...).
- Adaptation modules establish the schema required for that augmentation.

<table>
<thead>
<tr>
<th>server module: device 1</th>
<th>server module: device 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>coordination module: device 1</td>
<td>coordination module: device 2</td>
</tr>
<tr>
<td>client module: device 1 specific content</td>
<td>client module: content</td>
</tr>
<tr>
<td>adaptation module:device 1</td>
<td>adaptation module:device 2</td>
</tr>
</tbody>
</table>

T-Systems

-öffentlich-

Dr. H.-W. Sehring / A Modular Microservice Architecture for Multi-everything Content Management
CCM Microservices for Multi-everything Content Management. Personalization.

- Personalization according to the basic personalization pattern of CCMSs.
- Additionally, a server module to access plain content for the non-personalized case can be added.
- Alternatively, there can be one single server module, and a subordinate coordination module decides whether to show plain content or that for a specific target group.

server module: plain content

server module: personalized content

coordination module: personalization

adaptation module

client module 1: plain content

client module 2: personalized content
SUMMARY AND OUTLOOK.
Summary.

- Basic content management is well established for content creation and delivery.
- Additional requirements are emerging from application domains, especially marketing and sales, as well as knowledge representation.
- Content management systems are no longer isolated solutions.

- The technical answers to domain requirements are similar: multi-everything. All deal with
  - variance of representations and
  - adaptations of content.

- Concept-oriented content management is an approach that fits well into modern software engineering approaches – adaptable, non-monolithic, open, dynamic, ...
- By its nature it allows to compose components for contemporary requirements.
Outlook.

- So far CCMSs have only been applied in scientific contexts, not for commercial projects (customers). Such applications are yet to come.
- It might be interesting to look into pattern library for CCMS configurations that maps requirements to typical module constellations. This should be interesting not only for CCMSs, but for CMSs in general.
- This way some of the multi-* features should be unified. At least the ability to compose multi-* features should be further examined and manifested in patterns.
- One future research direction will be more support for marketing automation:
  - Feedback: tracking etc. can be applied for A/B testing, personalization, detection of conversion blockers, ...
  - User journeys exceeding the CCMS should be tracked to make use of that knowledge.
  - CCMS evolution might be enhanced if conversions can be detected. Eventually they should automatically adapt on that occasion (task completed, context should change).
Thank you for your attention!

Dr. Hans-Werner Sehring
Principal Software Architect
T-Systems Multimedia Solutions GmbH
Address Binderstraße 26, 20146 Hamburg, Germany
Contacts Phone +49 (40) 570062 8049
E-mail hans-werner.sehring@t-systems.com

http://www.linkedin.com/in/hwsehring
https://www.xing.com/profile/HansWerner_Sehring