Some remarks from an industrial partner perspective

11 November, 2009, General Assembly, Rome

Stefane Fermigier, Nuxeo

IKS is co-funded by the European Union and develops new technology for intelligent content management

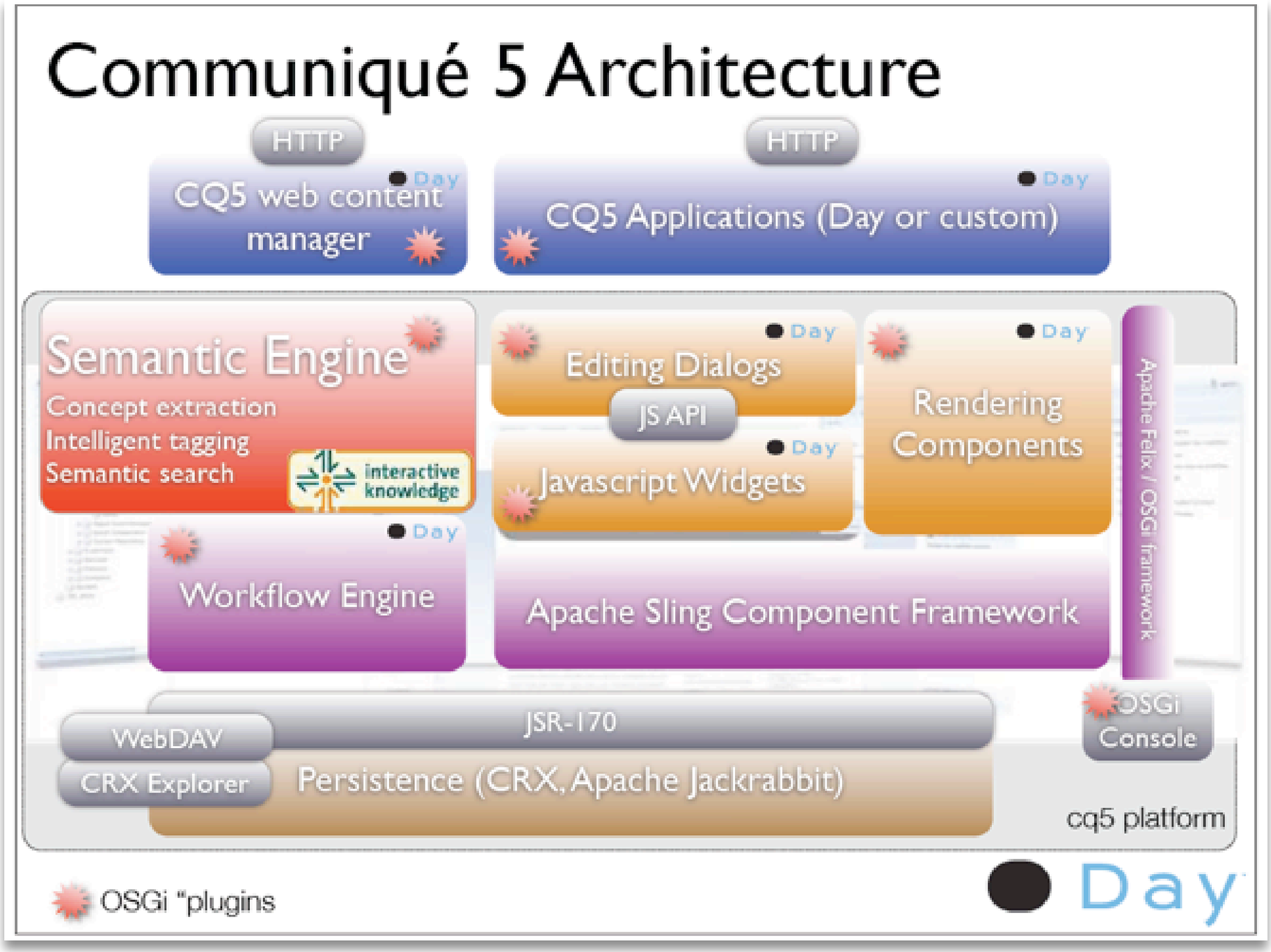
Use Cases / Requirements Gathering

- A first list of requirements has been collected
- We need to double check that we're not missing anything important
- Some of the current functional requirements are still vague
  - "We need to be able to issue SPARQL queries"
  - "We need to provide reasoning services"
- Should derive from actual, concrete use / business cases
- What are the leaders doing? Where's the market moving?
  - Thu/Fri workshop should tell us more
  - Need executive level stories, à la SWEO use cases (w3c.org)

The IKS Stack vs. modular services

- The stack should stand on its own (from storage to UI) for demo purposes
- Should also be breakable into atomic components / services to enhance existing CMS without major disruption
- Focus should be on the “semantic” (middle) part
  - Ex: Nuxeo has its own repository (can be enhanced to provide additional services needed by IKS, e.g. a triple store), its own GUIs (same)
  - Industrial partners should provide early feedback on how they will integrate the stack in their products

Example (Day)


Communiqué 5 Architecture
HTTP
CQ5 web content manager
Day
HTTP
CQ5 Applications (Day or custom)
Day
Semantic Engine
Concept extraction
Intelligent tagging
Semantic search
interactive knowledge
Day
Editing Dialogs
JS API
Day
Javascript Widgets
Day
Rendering Components
Apache Felix / OSGi framework
Day
Workflow Engine
Apache Sling Component Framework
WebDAV
CRX Explorer
JSR-170
Persistence (CRX, Apache Jackrabbit)
OSGi Console
cq5 platform
OSGi "plugins
Day

Services vs. libraries

- Not all the partners are Java based, some are
- Java libraries vs. language- (actually, platform-) agnostic services (HTTP, probably RESTful, with XML or JSON payloads)
- Ex: CMIS
  - Can be accessed as a web service (via RESTful, atompub interface, or SOAP)
  - The Chemistry project (Apache, led by Day and Nuxeo) also provides a Java API conforming to the CMIS model
  - Can be used for both in-process and inter-process API calls
- Integration / extension points must be clearly identified
  - Ex: OSGi, UIMA, CMIS, JSON/HTTP, SCA, AMQP, rule engine, scripting...

Buy or Build?

- Will the IKS developers only provide “glue” on top of existing open source technologies?
- Or contribute to existing OSS projects?
  - Is there a need for deep algorithmic work? Which ones?
- We should also keep our options open wrt which specific implementation we will eventually choose
  - Ex: “triple store” vs. “Jena” or “Sesame”
  - But not at the expense of another layer of useless indirection
  - I expect everyone will want a different UI

Non-functional, business-critical requirements (1/1)

- Make sure that customer value creation drives the use cases
- Quality
  - Components need to be production-ready when done
  - Solution: common coding guidelines, lots of tests, continuous integration, external scrutiny (=> low barrier to entry for external observers)
  - TCK (compliance tests) for alternative components?
- Performance / scalability
  - Ensure that the stack can manage common workloads for our customer projects, not just “toy” datasets (some customers, e.g. AFP, can provide real datasets)
  - Special attention to interprocess communication (e.g. HTTP calls)

Non-functional, business-critical requirements (2/2)

- **Licensing**
  - Make sure that both open source and proprietary vendors can benefit from the stack
  - Apache or BSD licensed components preferred, LGPL if necessary, GPL excluded
- **HR / training**
  - Make sure that the IKS stack can be used by Joe Average Programmer, not just PhDs in semantic technologies (or Hindley-Milner type systems, etc.)
  - Same for maintenance
- **Speaking of which: who will support the stack once the project is over?**
  - 80% of the effort on a project is spent on maintenance

Landscape is moving

- GGG / linked open data movement gaining steam
  - Business opportunities
  - Technical challenges (sheer amount of data / knowledge)
- Twitter
- NOSQL movement unknown 6 months ago, now has 50+ products
  - Some of them are related to what we do (e.g. graph and document databases)
  - Ex: neo4j, Cassandra, CouchDB, MongoDB, Hadoop
- New languages (ex: Scala, Clojure...)
- Multi-core / multi-threaded computing
  - New (or forgotten) programming paradigms becoming fashionable: MapReduce, functional programming, actors...

Planning

- According to planning, academic developers will work on the stack in 2010, and industrial partner start testing it in 2011 (or late 2010)
  - That's waterfall, and it's bound to fail
  - We need to be able to provide feedback much earlier
- When will coding begin?
  - Even without a clear global vision, it might be useful to start working on some useful subproject (ex: semantic search) to get the project moving
  - We also need to match our workload to our actual capability
  - Focus on low-hanging fruits and providing business value continuously (cf. Scrum, Kanban)

Conclusion / Recommendations

- Need for high level business stories (not just use cases) to sell the ROI both to end customers and to CMS providers
- Get quickly started on a (throwaway) prototype stack built with existing bricks and duct tape
- Don’t wait for an “alpha” (06/2010) version to start collecting feedback
- Focus on highest value stories (e.g. semantic search) to showcase delivered value
- Set up collaborative development process / framework (code repository, issue tracker, continuous integration) quickly