AT4AM

Authoring Tool for Amendments at the European Parliament

About me

- Philip Luppens
- Consultant at SuperMushi
- Enterprise Java + more exotic solutions
- Working at the European Parliament for 5 years
- Various projects (VISEP, AT4AM, DST, ...)
- Apache Software Foundation member and Struts PMC

Disclaimer

I’m a technical guy.

So this is going to be technical.

DON’T PANIC.

Feel free to interrupt me.

Let’s have a look at how things are working at the European Parliament.

Workflow

- Legislative proposals
- Assigned to committees for reading (Main and Opinion)
- Committee members propose amendments, ‘gathered’ by secretariat
- Voted upon
- Consolidated into new proposal
- Up to three times
- Plenary (free for all)

This is a simplified version.

Engage time machine.

Before 2008

- Microsoft Word
  - Incoming proposal in Word
  - Distributed as Word file
  - DOC EP (set of macros with ‘hidden’ tags)
  - Room for errors (location, diffing, ...)
  - Tedious, manual
  - Deadline was ‘more like a guideline’

We can do better. IT should be about making lives easier, not harder.

After 2008

- E-Parliament program
- First project: AT4AM
- Massive scope

- User requirements not clear
- Environment was ‘hostile’ (no trust, ‘we always did it this way’, ‘it works’, ...)
- Slightly less massive scope

Pick the right fights.

The beginning

- Evaluated several options: rich client, fat client, thin client
- Settled pretty quickly on web
    - Easier development
    - Easier distribution and upgrade path
- Two-tier solution
- And a huge reduction in scope: do one thing, do it well
- What comes in, must go out: MS Word

Microsoft Word.

Microsoft Word

- Extract structure from Word
  - Other way around. Massive loss of information.
- Build up tree model and store fragments in the database
- Align translations
  - Not as easy as it sounds
  - Developed tools to assist
- Cavalry

18

XML

- Another unit already did transformation from Word to SGML
- Developed a custom, Word-like representation schema
- KISS
- Tailored
- XSD is approx. 200 lines
- DEMO

XML in AT4AM

- Word to XML: loss of information
- Perhaps too simplistic
    - Not self containing
        - But size does matter
    - Structure and display only (no metadata)
- Not a standard
- Needed transformation into HTML before display

The work begins.

Backend

- Java stack
  - Spring, Hibernate, Struts 2, Quartz
  - Well know and proven
- Oracle DB
  - Well known
  - Truth be told: it is fast
  - Not perfect

22

Frontend?

- Lots of discussion
- No Java applets, no Flash
- So only Javascript remains ...
- Only in extremely limited circumstances
- Limited experience
- It’s a mess, but Javascript frameworks help (a bit)

So, what did we choose?

Google Web Toolkit.
It rocks.

Google Web Toolkit

- Lots of experience
  - But not universally positive
- Fantastic idea
  - Java compiled to Javascript
  - Reuse of tools and talent
  - Cross browser
  - Very performant

Don’t lose yourself in technical details.

Keep in mind ...

- No-one cares about structure or backend
  - Best tool for the job
  - Within given constraints
- Usability of utmost concern
- Better than Word
  - Otherwise, why would anyone use it?
  - So you need something extra ...

Kicking Word around

- Word-like representation, but faster
- No room for error
- Location determination
    - Eg. Article 4 - paragraph 2 - point 2 - subpoint 4a (new)
    - XML structure is a tree, so ... easy(ish)!
- Automatic diff-ing and spellcheck
- Collaboration

Wishes

- Display Word document
- Draft amendment: modification, creation, deletion
- Movement
- Loading of previous amendments
- Automatic diff-ing and spellcheck
- Tabling, withdrawing
- Export to Word and XML

Export to Microsoft Word

- Export to Word is extremely important
  - Still used by big parts of the EP
  - Ensure rest of the workflows keeps running
  - Allows easy sharing readonly copy
  - Final fallback for users

Export to XML

- Export to Akoma Ntoso
  - Proof of concept
  - Gives other systems the chance to adapt and prepare
- Hard to get it right
  - Freemarker allows for quick and easy alteration
  - Good enough to sell the idea
  - But text is not XML, so correctness relies on previous steps

Ensure XML validity

- Content editing via TinyMCE
  - ‘Allowed tags’
  - Cleanup and validation
- Backend used to do cleanup via JTidy
  - But it’s dangerous: rules alter markup from diff-ing
- Trying to be even more strict in the open source version

A prototype - my kingdom for a prototype

Prototype

- Based on Observer pattern
- XML pushed as fragments from the DB
- It worked!
- But performance was pretty bad
  - 20 page document in 10 seconds
  - Memory usage through the roof

If at first you don’t succeed ...

Performance: the breakthrough

- Google I/O session
- Overlay types
  - Transform XML into HTML
  - Use .setInnerHTML
  - Overlay existing DOM tree (no reflows/repaints) and attach listeners
- Increased performance 100x and reduced memory 20x

Editor

- Overlaying
- Memoizing
- Fast lookup tradeoffs
- Tag hierarchy modeled as inheritance system
- Marker interfaces mostly for location determination
- Injection via element and location
- DEMO

38

Amendment Bundles

- Multiple amendments combined into one
- Only on a single element and its children
- Contains both original content and amendments
- DEMO

Footnotes

- Part of the amendment
- Modeled as a list of child objects
- Simple validation
- Simple state: creation, modification and deletion

Tables

- Table visibility
- Table editor
- Tables are a bit of a problem
    - Technically, columns, cells and rows are
- Virtual location extensions
    - It’s a hack

Injection

- Three types
  - Simple: foo:233
  - New collection: foo:233:3
  - Existing collection: foo:233:+3
- Amendments might be out of order
- Amendment can be loaded from different sources
  - So amendments might have the same injection location

Diff-ing

- Both client and server-side
- Readability versus correctness
- Markup versus content

- Diff-ing is hard.
  - Several attempts
- Turned into a service.

Versioning

- Introduced late into the project (only upon DST arrival)
- Amendment version trees (joined using graphID or revisionID)
- Primary key on amendment defines one version
- Keeps track of sibling and total version count

44

Versioning problems

- Hard
  - Push it on the storage engine if you can!
- Complex for queries
- Storage space
- UI issues
- Versions are not linear: multiple siblings are possible
  - But rarely wanted

45

Internationalization gotchas

- 23 languages and counting
  - GWT resource bundles to the rescue
  - But watch your permutations
- UI language and source text language not always the same
  - eg. German UI and English source text
  - ... and ‘panache’ amendments in French

Performance

- Static resources
  - Proper E-Tags, Cache headers, minification, gzip, ...
  - WRO4J, GWT
- Garbage collection
  - Watch your JVM
- Use JMeter with proxy recording
- Client: IE is dog slow

Thanks, IE.

I rarely consider violence as an appropriate action, but ...

Internet Explorer: the horror

- IE is the only target browser
  - Lots of legacy software
  - The only ‘secure’ option
- Slow, bug-ridden, horrible piece of software
- No profiling or decent debugger
- On IE7 until 1.5 years ago

Once again, GWT saves the day

- Everyone developing in Firefox
    - Profiling & inspection via Firebug & Java via OOPHM
- IE only for verification
- It kept us (relatively) sane

All is well that ends well ...

Users love it!

- Tailored dashboard with upcoming dossiers
- Filtering system
- Amendment draft time decreased massively
- No structural errors possible, so (almost) no returns
- People are going wild after being involved in the testing rounds
- Credibility and trust gained

Team

- Currently 2 developers AT4AM, 1 DST, 1 architect, 1 analyst, 1 PL, 1 PM
- Total of 16 people in 4,5 years
- Experience and talent less relevant
- Not exactly rocket science
- Attitude and social skills
- Passion, willingness to learn
- Good atmosphere

Planning for world domination.

Spending karma

- ‘Trust us’ (we know what we’re doing)
- Full freedom to design
- ‘Unacceptable’ proposals
  - Ask permission later
  - Hierarchy is a powerful thing
- But you need to deliver: lose your users, all is lost

Three’s a crowd.

Stepping up the game

- Increasing scope
  - Verification (DST)
  - Translation (CAT4TRAD)
- Three tier solution
  - Webservices all the way
  - Service4AM

58

DST

- Verification of amendments before translation
- Specialized personnel (60 lawyer-linguists, 1 to 5 per language)
- 1 day window to review amendments (up to 3000, batched per 300)
- Not necessarily 8 hours
- Word oriented
- Issues with locking & merging of documents

59

Not the easiest clients. They have some pretty strict requirements.

A new hope

- Credibility and freedom
- Full integration with AT4AM
  - Hence the three tier architecture
- Points of interest
  - Number of amendments much larger
  - Collaboration
  - Track-changes, comments, locking

61

DST Demo

And of course, with new requirements ...

Once again, performance

- Frontend
    - UI-Binder, lazy rendering
- Backend
    - Complex domain model
    - DB suffering (joins & clobs)
    - Querying is hard

“Thou shalt not do unnecessary work.”

Caching

- Hibernate caching (Level 1 & 2)
- Spring method cache
    - Cache DTO by (primary) key
    - Query returns only (primary) key
    - Expire key upon change
        - Hibernate interceptors
    - Massive gains (>1.000.000 amendments/hour)

Ok, that was pretty technical.

So. Here’s a picture of a cat.

Cute cat

Bill Suicine

Collaboration

- DST first
    - Needed for lock broadcasting
    - Extended to push updates to amendments
    - Extended to push highlights/comments
- HTML push is hard, so use the right tools
    - Atmosphere
- DEMO

Collaboration AT4AM-DST

- Cross application broadcasting
    - Assistance request during amendment creation
- Via Service4AM
    - Pushes as serializable message on MQ
    - AT4AM and DST both listening on MQ
    - Atmosphere responsible for distributing

Development

Persistence

- Oracle
    - When all you have is a hammer ...
- Alternative solutions: document DBs
    - Much more natural
    - But less experience
    - Hard to sell to DevOps and support teams

73

Methodology

Test Driven Development

Nope.

Surprise Driven Development

Development methodology

- Build prototypes fast
- Plan for change
- Rapid turnaround
- Feedback loop
- Assume other projects will be faster/later/missing

Testing

- Unit testing
    - Yeah
- Integration
- Regressions
- Turns out, it’s not that bad: everyone knows everything
    - Makes estimations more solid
    - Allows for far bigger refactorings and modifications

You’re never alone.

Integrating other systems

- Variety of systems
  - Both legacy and new projects
  - Public sector, so delivery estimations are hard
    - Prepare for the worst
- Grand vision toward SOA
  - Webservices, ESBs, MQ
  - Communication and cooperation is hard

When things go wrong at 4 am.

(haha.)

When things go wrong

- Assume things will go down
- MQs to the rescue: async processing and persistence
- Prepare yourself

- Admin interface
- Console
- Backups
- Monitor

Where are we after 2 years in production?

Amendment statistics

- Approx. 196.000 amendments
- Approx. 100.000 in the last 9 months
- Top months 2012 May (18.632), June (21.124), July (17.534)
- Approx. 3% created during weekends
- Approx. 700 created between 1.00-6.00, 28% between 10.00-12.00
- Approx. 81% during office hours (9.00-17.00)

More statistics

- 15% deleted, 8% candidate
- 1.3% in plenary (2.820)
- 325 amendments by 1 author, 28 average
- 86% MEPs, 41% assistants, 100% others
- 45 deadlines on one moment, 189 in a month

Data mining

- Not much
  - BI reports
  - Simple statistics
  - Several hooks into eventing system
  - Statistics page for ‘management’, forecast page for expected workload
- No real content processing
- Room for growth and ideas: open source

Closing remarks

- Focus on one thing
- Small steps
- Feedback loop
- Visible things first
- Simplicity
- Performance
- Use the hierarchy to your advantage

Q&A

Thanks!

Development

- Source control (since 2 years on SVN)
  - Git please!
- Environments (DV, PP, PR, IT, FP, ...)
- Continuous integration & build
  - First Ant, then Maven
  - One click to deploy a revision from SVN into DV

Environment

- AT4AM, DST, WS4AM and UTIL4AM
- DV, PP, PR, IT, FP
- 8 core virtualized machine
- SunOs 5.10
  - Watch your file encodings
- Tomcat 6.0.18, Oracle 10g, HornetQ 2.2.14
- JVM 1-2 GB heap

Some performance tricks

- Heavy document/DTO caching
- Future<?>
    - Parallel
    - Fault tolerance
    - Awesome

Amendment

- Two columns
    - Original text, amendment text
- Each column is split up in a list
    - Allows for correct diffing in case of bundles
    - But stored as a single clob column in DB
- Text and diffing
    - So 4 DB columns

But we can do better ...

- Performance: injection time for amendments
    - Crazy solutions: serverside rendering, JIT replacements
- UI
    - Less is more
    - Apple
- Usability
    - Some things take you by surprise: font size, contrast, resolution