The update

• The move to MKIV is more than supporting an engine.
• It is a complete rewrite (pruning, extending, cleaning).
• Although somewhat crippled by the fact that we want to remain compatible.
• But sometimes we sacrifice compatibility by getting rid of old stuff.

The current state

• A lot of work, more than I had thought, so it takes longer.
• Most \TeX\ code is done (some structure and column code left).
• New namespaces and helpers mostly done, but will be checked for constency.
After the cleanup

What is there to do

• Some code might become generalized (also depends on others).
• Layer and positioning code might get a more extensive LUA and XML interface.
• Structure-related code will support setups (some already in place).
• New page builder variants will be explored (anyway more column support and floats).
• Math domains cq. dictionaries (basics already in place, just needs time).
• Math list optimization (pet project).
• Generate dependency trees (easier now) and more consistent code loading order.
• All error messages need checking (some gone, some not yet translated).
• Update all XML definitions (work in progress, also relates to wiki).
• Optimize positioning system (a bit more powerful now, but also more resources).
• More support for CSS-like styling (makes it easier to share code).
• Modules (especially those for tracing) need to be normalized.
• Some styles (mostly private presentation styles) need to be fixed.
• Pick up the 'lean and mean' CONTEXT variant project.
• Now that we have more code isolated, we can define an API.
• Some manuals need to be updated (most still apply).
What I have to keep in mind

• What is handy for me is not always handy for all users.

• But nevertheless there will be new things.

• Elements of our processing framework will show up in the distribution.

• It's just more convenient to have one installation for related things.

• This is also why support for databases has been added recently.

• Running (blocking) TEX jobs needs special treatment (ticket management).

• It makes sense to use the well developed TDS infrastructure.
• Rewriting the code base leads to bugs but these are often resolved quickly (indeed by Wolfgang).
• Following the mailing list helps and nowadays the wiki is adapted close to realtime (coordinated by Sietse).
• Changes in standards and related tools are supported and followed by those who depend on them (ask Peter).
• Sometimes users have demands and these end up as extensions to existing mechanisms (Aditya’s elastic modules).
• Issues with platforms are often quickly dealt with (if Luigi doesn’t know it . . .).
• And of course I add new things driven by projects, challenges (and an occasional stack of new CD’s).
• New releases (and betas) are checked against a growing set of test files (Lukas mails a report after each update).
Breskens 2012

After the cleanup

We started long ago with what ended up as MKII and MPII. We currently have MKIV and MPIV. It has some features that we tag as MKVI. Recently MKIX and MKXI were introduced.

Examples: MKIV, MKVI, MKIX, MKXI

So, what should MKIC provide:

What I'm working on:

- Proper dependency chain so that we can make small dedicated formats.
- Support for typesetting from databases (text, graphics).
- Next iteration if (job) tickets processing system cq. framework.