Panda Workshop at UTA

DEFT

Maxim Potekhin
BNL

Sep 03, 2013

About this presentation

- We have a number of expert developers coming on board for the next few months to support the DEFT effort
- We should not take for granted that even with current detailed documentation all aspects of the system will be clear to everyone involved from day one, so it may be a good idea to revisit basic design principles and post links to documentation
- I meant for these slides to be a "quick start" guide for anyone involved in the project, backed up by our (pretty good) set of TWiki pages
- …and of course I also need to report on the status and immediate work plan for DEFT

DEFT in a nutshell

- DEFT is the front-end for the new Production System, known as Prodsys2
- It incorporates a new model, the Meta-Task, which was designed to reflect the actual mode of operation of physics working groups that emerged in recent years, and to provide adequate support of their workflow
- The Meta-Task is essentially a DAG representation of the workflow, garnished with Panda-specific attributes
- The other half of Prodsys2 – JEDI – picks up database records created by DEFT and formulates job definitions for each task, to be processed in Panda
- Note that JEDI is Meta-Task-Agnostic by design. It translates task request (DB records) into collections of jobs.

DEFT components

- DEFT has two major components, named "deft-core" and "deft-ui"
- **deft-core** is a collection of classes and functions that allow the Meta-Task definitions, created by managers in XML format, human-readable and industry standard, to be stored in RDBMS. The other (symmetric) part of this functionality is the ability to serialize the Meta-Task object from its RDBMS representation, into XML
- **deft-core** also includes the following important functionality:
  - Creation of Meta-Task based on a pre-existing template
  - Ability to use existing Meta-Tasks as templates (the "copy" function)
  - Acting as a State Machine, by keeping track of the DAG state, and controlling submissions of tasks to JEDI for further processing. Tasks can be put on hold, cancelled and activated by managers.
  - Text-based monitoring and editing of attributes

A bit of DEFT history and philosophy

- From the very beginning, DEFT was designed based solely on the user (manager) requirements. in particular a few interviews with Wolfgang Ehrenfeld and later with Nurcan Ozturk. There is nothing more and nothing less in the design.
- The ad hoc system now in use, based on Excel spreadsheet, was a quick solution to the problem of handling Meta-Tasks, and it proved difficult to scale and maintain. The essential part of transition to Prodsys2 is that the dependencies among the tasks forming a workflow are made explicit in XML as opposed to being contained in a suite of obscure scripts.
- DEFT is designed to have a command line interface (CLI) and a Web UI that are identical in terms of functionality. This is essential from operations perspective.

DEFT Documentation

- The root for all of Prodsys2 TWiki documentation is this: https://twiki.cern.ch/twiki/bin/viewauth/AtlasComputing/ProdSys
- The DEFT Web UI info is here: https://twiki.cern.ch/twiki/bin/viewauth/AtlasComputing/DeftGui
- All documentation is updated reasonably often, and contributing team members are certainly encouraged to add or modify the content as required by the project
- Note that the CERN-based TWiki was recently refactored, most links appear to work but in case some don't, please be patient and let me know of any problems.

The DEFT SVN

- All code is actively maintained in SVN. The layout of the latter was modified in August based on suggestions from team members.
- **https://svnweb.cern.ch/trac/panda/browser/panda-deft** is the root for both deft-core and deft-ui
- What's not in SVN: the database credentials, coded into a Python function. Contact me for details. This will be further improved to allow a quick switch between ADCR and INT databases.

The DEFT database

- Database schemas are being brought in line with what's currently the API of JEDI-α and ATKR adapter. Two main schemas are kept up-to-date in the SVN (*.sql).
- These are the tables currently used:
  - META
  - TASK
  - DATASET
  - COMM
- The latter being the semaphore container for communication with JEDI
- Still missing – the job template dictionary, currently implemented as runtime in the AKTR adapter

A note on deft-core

- The code has been in place and working for a while
- Not much coding still to do, mainly expansion of the database schemas
- Does require a good knowledge of the Production System to move forward and maintain, can be best maintained by experts such as Sasha

DEFT UI: the platform

- We chose Django for multiple reasons which we won't repeat here
- A development/alpha testing machine was quickly setup at CERN after the June S&C meeting (thanks to Mr.Baranov) and the support has been quite good
- Django allows for very quick creation of template-based HTML content delivery, and is also easily instrumented with JSON serialization functionality. The assumption is that for a while these two methods will co-exist in development.
- A simple Django application has been in place for a while, serving both JSON and HTML .
- HTML: right now, no effort is put into making it pretty as it is primarily a development tool allowing for quick and dirty data visualization.

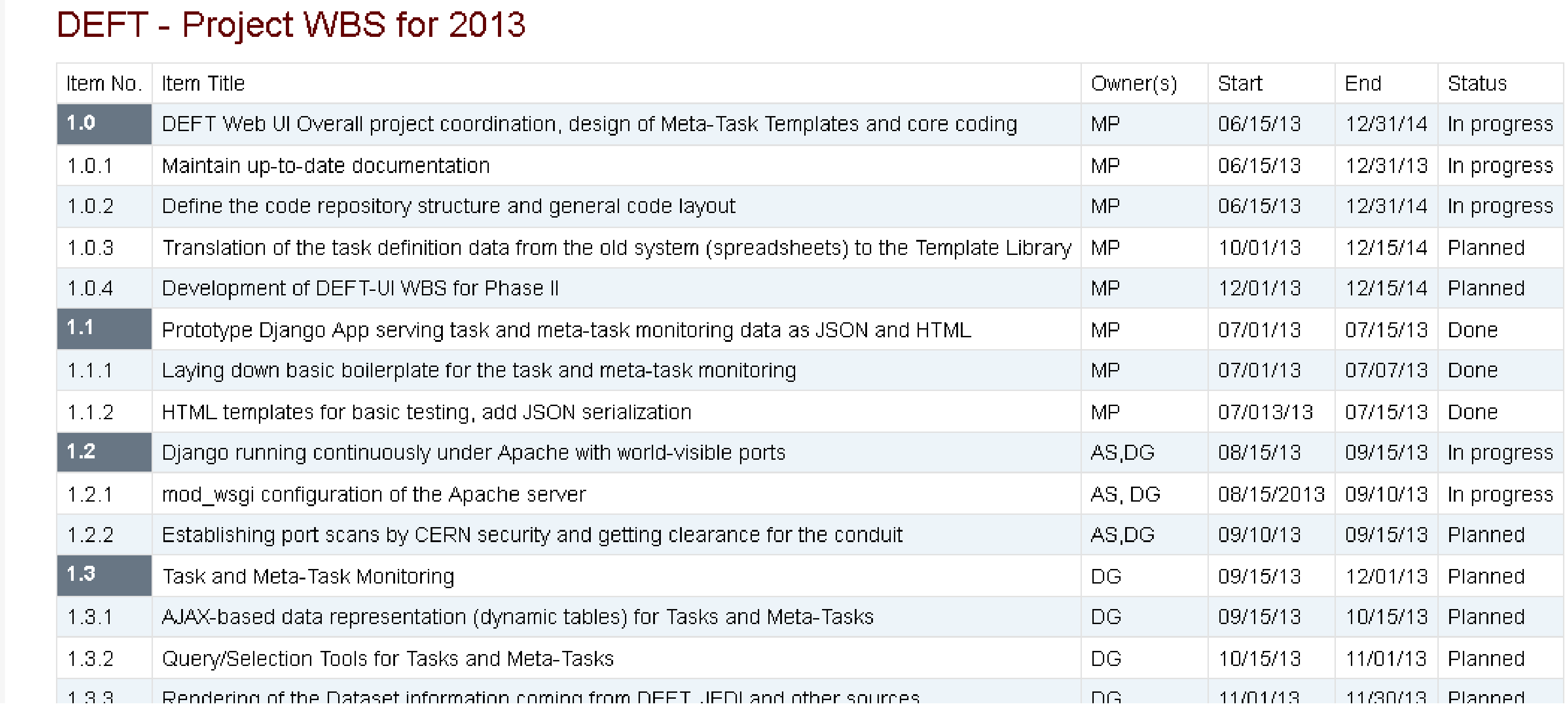
DEFT UI: basic functionality

- Record of our design documents
  https://twiki.cern.ch/twiki/bin/viewauth/Atlas/DeftGui#Views_Screens_Functions_and_gene
- In a nutshell, the very basic functionality is covered by:
  - Two monitoring pages, for tasks and Meta-Tasks, with obvious cross-links between the two as needed
  - Template library: derivation of Meta-Tasks from prefab examples and editing/adjusting their parameters as needed (w/o change of the Meta-Task topology). Closely related to this is Meta-Task cloning functionality, which can be co-located on the same page
  - Approval and Control: administrative page reserved for managers
- The above list covers the minimal but complete functionality of the DEFT end-user and manager interface
- The good news is that it's only 4 screens total, which is not a large number. Works can be split nicely using Django modular organization. We will need a reasonable navigation bar etc.

- Administration of user access is included in Django, can be left for later will we finish alpha-testing.

DEFT UI: the WBS

- See detailed WBS at: https://twiki.cern.ch/twiki/bin/viewauth/AtlasComputing/DeftWbs



DEFT - Project WBS for 2013

Item No.	Item Title	Owner(s)	Start	End	Status
1.0	DEFT Web UI Overall project coordination, design of Meta-Task Templates and core coding	MP	06/15/13	12/31/14	In progress
1.0.1	Maintain up-to-date documentation	MP	06/15/13	12/31/13	In progress
1.0.2	Define the code repository structure and general code layout	MP	06/15/13	12/31/14	In progress
1.0.3	Translation of the task definition data from the old system (spreadsheets) to the Template Library	MP	10/01/13	12/15/14	Planned
1.0.4	Development of DEFT-UI WBS for Phase II	MP	12/01/13	12/15/14	Planned
1.1	Prototype Django App serving task and meta-task monitoring data as JSON and HTML	MP	07/01/13	07/15/13	Done
1.1.1	Laying down basic boilerplate for the task and meta-task monitoring	MP	07/01/13	07/07/13	Done
1.1.2	HTML templates for basic testing, add JSON serialization	MP	07/013/13	07/15/13	Done
1.2	Django running continuously under Apache with world-visible ports	AS,DG	08/15/13	09/15/13	In progress
1.2.1	mod_wsgi configuration of the Apache server	AS, DG	08/15/2013	09/10/13	In progress
1.2.2	Establishing port scans by CERN security and getting clearance for the conduit	AS,DG	09/10/13	09/15/13	Planned
1.3	Task and Meta-Task Monitoring	DG	09/15/13	12/01/13	Planned
1.3.1	AJAX-based data representation (dynamic tables) for Tasks and Meta-Tasks	DG	09/15/13	10/15/13	Planned
1.3.2	Query/Selection Tools for Tasks and Meta-Tasks	DG	10/15/13	11/01/13	Planned
1.3.3	Rendering of the Dataset information coming from DEFT, JEDI and other sources	DG	11/01/13	11/30/13	Planned

DEFT UI: the game plan

- Dmitry and Stavro will form the core development force for the UI project in the Fall of 2013
- Alden will contribute to specific areas of the project such as handling of restricted pages and task control functions
- WBS, as detailed as it is, is not the same as design documentation. We have a good foundation for that in our TWiki pages which will be soon enhanced with layouts, mockups, graphics etc. Proper docs and communication will help the progress of the project.
- The idea is to compliment the knowledge of business logic (Sasha+Maxim) with a competent Django development team (Stavro+Dmitry).
- The Apache service will be started soon, but even before that, anyone can contribute to the project using the development server

DEFT UI: simplicity and phased approach

- Keep it simple at every level, from basic code organization to the Django module layout.
- Don't worry *too much* about visual appearance at this stage, prioritize functionality over graphics
- I propose a staged approach to the UI development
  - Full functionality must be there in late 2013 – **stage one**.
  - We need to be able to demo the app to the managers in a month or two
  - This needs to be the focus (even if no full AJAX support exists in the first stages)
  - This buys extra time to AJAXify/prettify/jQuerify the application in Nov-Feb 2014: **stage two**. Note that some work will be done in parallel with stage one.

DEFT UI: conclusions and outlook

- Even with resources reduced as compared to our original plan, the project can and will be done on schedule with proper planning and minimalistic approach to the UI
- Communications are of essence, and so far have been satisfactory, with shared and extensive documentation as well
- We will definitely have a weekly phone conference for developers (MP, DG,SG) and bi-weekly reports to be given to AK and other managers
- WBS is in place for the rest of 2013, stage II WBS (2014) will be delivered later this year.