TripCom Implementation Plan and Strategy (V2)

Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project

Component	Supervision	Phase 1	Phase 2	Phase 3
System Integration	WP6/TUW	-	T6.5 (19-24)	T6.7 (29-33)
Triple Space API	WP3/FUB	T3.2 (6-12)	-	T3.7 (29-33)
Triple Store Adapter	WP1/ONTO	T1.3 (7-12)	T1.6 (13-18)	-
Triple Space Manager	WP2/FUB	T2.3 (7-12)	T2.6 (19-23)	T2.8 (24-28)
Security Manager	WP5/CEFRIEL	-	T5.3 (19-24)	T5.6 (29-36)
Mediation Manager	WP4/TID	-	T4.5 (19-24)	T4.6 (28-36)
Metadata Manager	WP2/LFUI	-	T2.6 (19-23)	-
Query Processor	WP3/PROFIUM	-	T3.5 (19-23)	T3.6 (24-28)
				T3.7 (29-33)
Transaction Manager	WP3/FUB	-	T3.5 (19-23)	T3.7 (29-33)
Distribution Manager	WP2/FUB,ONTO	-	T1.7 (19-24)	T2.8 (24-28)
		-	T2.6 (19-23)	
Web Service Invocation	WP4/NUIG	-	T4.2 (13-24)	T4.6 (28-36)
Web Service Discovery	WP4/NUIG	-	T4.2 (13-24)	T4.6 (28-36)
Web Service Registry	WP4/TID	-	T4.3 (13-18)	-
Use Case 8A	WP8A/TID	-	T8A.5 (13-22)	T8A.9 (31-36)
Use Case 8B	WP8B/CEFRIEL	-	T8B.3 (16-22)	T8B.7 (31-36)

TripCom Implementation Plan

- Next level of detail to be specified by responsible partner
- Deadline for subcomponents/-functionalities
  - by Oct.12
- DERI-CVS
  - directory `ImplementationPlan`
  - `WP<n>_<component_name>.pdf`
- Next step:
  - **precisely specify interface to other components, i.e. to XVSM**
- Review by all partners
  - concerning dependencies
  - ***check for inconsistencies***
  - by Oct.22

Responsibilities of Partners

- Managing implementation work of assigned component
- Deliver component implementation and source code
  - by deadline
  - tested (unit test),
  - including documentation
    - Javadoc
    - Checkstyle

Test of Individual Component (Unit Test)

- Create test suite
  - definition of testcases
    - functional aspects
    - performance aspects (time measurements)
    - including scalability benchmarks
    - demonstrate components fitness wrt. Scalability
    - contribution to system scalability, as expressed by parameters
  - communicate to others
    - via integration platform
- Commitment/review of test cases by others
- Perform tests
  - > 60% coverage required
  - Communicate test results
    - achieved coverage
    - achieved performance & scalability parameters

Development Process: Interfaces

- Starting point: for each component:
  - specify input/output behaviour wrt. integration platform (XVSM)
  - component descriptions section 2.5 of D6.3, Interaction with other components:
    - *detailed structure of containers and tuples this component writes and reads from/to XVSM. Also, describe possible sub-spaces this component creates/uses*
- ***precisely*** *specify interface to other components, i.e. to XVSM*
  - in the form of tuples *(type_of_entry_1 entry_1, type_of_entry_2, entry_2, ...)*, where *type_of_entry* is a basis type (integer, string, ...) or at an abstraction level similar to the TS-API specification (Triple, Template, ...)

Development Process: Interfaces

- **+** precise description of parameters and types
- **+**
  - in case of write to XVSM (*out): which component is expected to read the tuple = target component(s)*
  - *in case of read to XVSM (in,rd): which component is expected to write the tuple = source component(s)*
  - *to be applied analogously to subscriptions*
- **+** representative scenarios of sequences of (XVSM-) operations
  - textual or (**preferably**)
  - UML sequence diagrams
- details to be specified later (*XVSM-API stage*)
  - container name, internal structure (entries), coordination type

Development Process: Interfaces

- check for inconsistencies
  - for each component:
    - identify inconsistency with interface specification of source and target component(s)
    - **+** suggested solution to solve conflicts
    - document in directory `ImplementationPlan, WP<n>_<component_name>_conflicts.pdf`
  - ***by Oct. 22***
- ***„bilateral“ contacts among WP's to resolve conflicts***
- *finally discuss and solve remaining inconsistencies in Skype conference of WP leaders (to be announced)*
- *(conflicts at XVSM-API stage to be documented, discussed, resolved using Sourceforge's issue tracking)*

Development Process based on Scrum

- One month implementation periods
  - focused effort
  - goals fixed for one month
  - implementation „sprints“
- Every month
  - list of features of component which will be implemented during next month:
    - todo-list (“backlog”), “sprint plan”
  - initial version defined by “component team”
    - WP Leader is responsible, proposes, coordinates
    - other partners concerned propose extensions/modifications
    - checked/confirmed by component team

Development Process

- No interference during one month
- Skype conf of all WP leaders at end of each sprint
  - Reaction, Feedback
    - what was done?
    - why something was not done?
    - problems, suggestions
  - for reaction on problems and changing requirements

Infrastructure

- sourceforge.net, berlios.de or similar service
  - simple to use, minimal maintenance costs
- vs. self maintained software and servers @DERI
  - needs (at least) one responsible person
  - more flexibility
  - needed tools (at least)
    - version control (e.g. *subversion*)
    - bug tracking (e.g. *trac*)
    - communication: forum, mailing lists, wiki

Tools

- Build system: *Maven*
  - automatic dependency management
  - allows to configure sub-projects and build them at once
  - uses some „best practice“ guidelines
  - every component ist a maven project
    - super-project for integration
- Unit tests: *JUnit, JCoverage*
- Check for coding style: *Checkstyle*
- Tools and format for documentation?
  - MS Office, OpenOffice, Latex, html, ...

Links

- maven: http://maven.apache.org
- subversion http://subversion.tigris.org
- junit http://www.junit.org
- checkstyle http://checkstyle.sourceforge.net
- trac http://trac.edgewall.org
- sourceforge: http://www.sourceforge.net
- berlios: http://www.berlios.de