CAS 756: Modelling and Metamodelling

Homework Assignment 1

Overview

- For this assignment you need to design a number of metamodels (DSLs) using the UML class diagram notation on paper
  - In some cases, you also need to design models conforming to these metamodels, using the UML object-diagram notation
- There are 3 DSLs/metamodels for you to produce.
- Attempt each question.
- Hand your answers in at the start of class on Friday 7 February (on paper!)

CONFERENCE DSL

Language Description

- Design a DSL for modelling conferences
- A conference runs over a number of days
- On every day, there are several talks organised in (potentially parallel) tracks
- There are breaks between tracks (e.g. for lunch, coffee etc.)
- Each track/break takes place in one room
- Each talk can be delivered by one ore more speakers
- Each talk has a pre-defined duration

Why?

- To ensure that the conference program is clash-free e.g.
  - Parallel tracks happen in different rooms
  - The total duration of the talks of a track does not exceed the duration of the track
  - Breaks don't overlap with tracks
- To generate booklets, web-pages etc. from the program in a consistent manner
  - Instead of maintaining them manually (risking inconsistency)

What now?

- Sketch a metamodel for the conference DSL using pen and paper
- Use only the UML class diagram syntax.
- If you identify any constraints that are needed to prevent invalid models from being produced, write them down in English.

SOFTWARE DISTRIBUTION DSL

Problem Description

- Software vendors need to build several bundles for different types of customers
- All these bundles are typically assembled from the same pool of components
  - Different bundles contain different subsets of these components
- Components have dependencies between them
  - e.g. if component C2 depends on component C1, then bundles that contain C2 must always also contain C1
- Exercise
  - Create a DSL for designing such bundles
  - Create a model that conforms to the DSL and exercises all its features at least once

Example

- You are a vendor of an Enterprise Resource Planning system implemented in Java that consists of several components
  - E.g. Sales, Warehouse, Payroll
- Each component consists of a number of JAR files
  - Components can share JARs
- The dependencies between your components are as shown in the next slide
- You wish to assemble different bundles for e.g.
  - Sole Traders: Core, Sales, CRM
  - Service Companies: Core, Payroll, CRM
  - Manufacturing Companies: All components excluding Real-Time Warehouse Analytics
  - Large Manufacturing Companies: All components

Why?

- You could write a packaging (e.g. shell, ANT, Gradle) script for each distribution manually however
  - It would be error-prone
  - They would contain a lot of duplication
  - They would be hard to maintain for a large set of components
- Using a domain-specific model
  - You can capture bundle configurations at an appropriate level of abstraction
  - You can perform checks for e.g.
    - components with cyclic dependencies
    - components/JARs that are not used in any products (obsolete?)
  - You can generate these packaging scripts automatically and they will be correct by construction
- This is how we **actually** produce all the different bundles available under the JARs tab of http://www.eclipse.org/epsilon/download/

What to do?

- Sketch a metamodel for this domain-specific language using the UML class diagram syntax.
- If you identify any constraints that are needed to prevent invalid models from being produced, write them down in English.
- BONUS: explore the Object Constraint Language, and use that to specify any constraints you identify.

RESEARCH PROJECT DSL

Language Description

- Research projects are typically conducted in a collaborative manner by a number of partners (universities, companies, charities etc.) and have a fixed duration (in months – e.g. 36 months)
- A project is split into a number of work-packages
- Each work-package has a start and an end month and is further broken down into more fine-grained tasks and deliverables
  - Tasks also have a start and an end month and each deliverable is due in a specific month
- Each partner declares how much effort (in person/months) they will allocate to each task

Why?

- Proposal documents contain several tables with overlapping information (screenshots in the following slides) e.g.
  - Effort per partner per task for a work-package
  - Effort per partner for the whole project
  - Table of deliverables for the whole project in chronological order
  - A Gantt chart that summarises the timeline of the project
- Unless these tables are generated from a common source (i.e. a model) they can become inconsistent with each other
  - e.g. a partner may change their effort for a task but forget to change the overall effort figure for the entire project
- Other consistency problems can also appear e.g.
  - Tasks that start before / end after the work-package in which they are contained
  - Deliverables that are due after their work-package ends
- This is how we **actually** write proposals for research projects

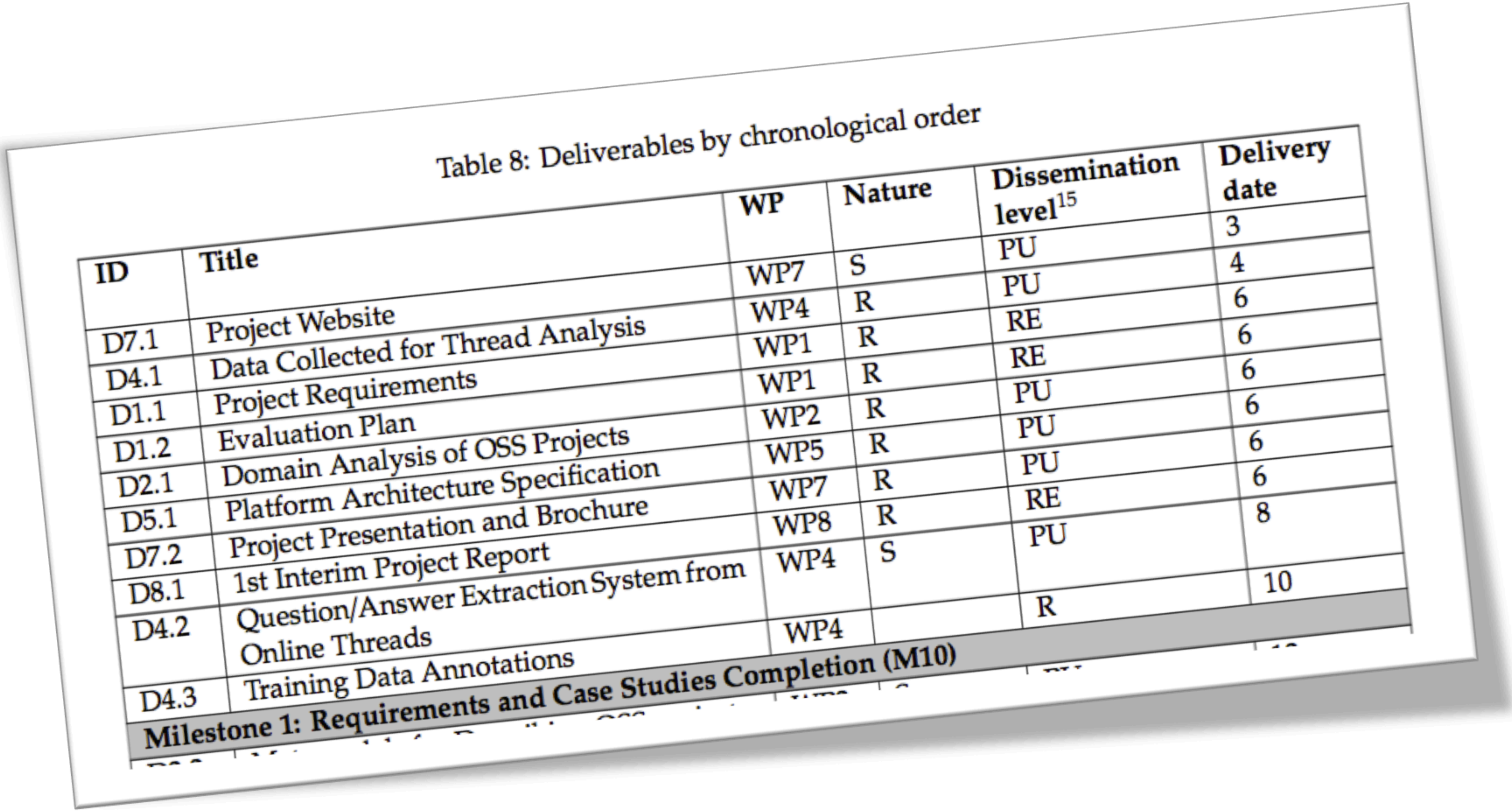

Table 8: Deliverables by chronological order

ID	Title	WP	Nature	Dissemination level[15]	Delivery date
D7.1	Project Website	WP7	S	PU	3
D4.1	Data Collected for Thread Analysis	WP4	R	PU	4
D1.1	Project Requirements	WP1	R	RE	6
D1.2	Evaluation Plan	WP1	R	RE	6
D2.1	Domain Analysis of OSS Projects	WP2	R	PU	6
D5.1	Platform Architecture Specification	WP5	R	PU	6
D7.2	Project Presentation and Brochure	WP7	R	PU	6
D8.1	1st Interim Project Report	WP8	R	RE	6
D4.2	Question/Answer Extraction System from Online Threads	WP4	S	PU	8
D4.3	Training Data Annotations	WP4		R	10
Milestone 1: Requirements and Case Studies Completion (M10)					

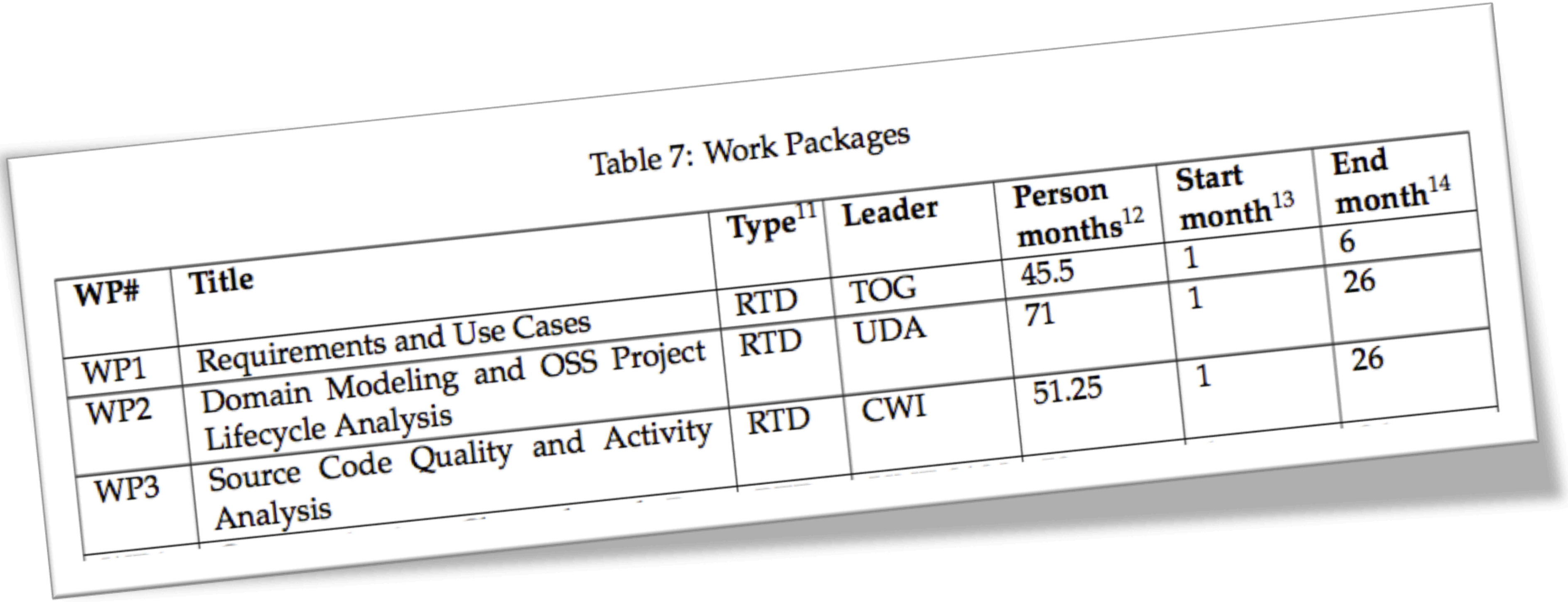

Table 7: Work Packages

WP#	Title	Type[11]	Leader	Person months[12]	Start month[13]	End month[14]
WP1	Requirements and Use Cases	RTD	TOG	45.5	1	6
WP2	Domain Modeling and OSS Project Lifecycle Analysis	RTD	UDA	71	1	26
WP3	Source Code Quality and Activity Analysis	RTD	CWI	51.25	1	26

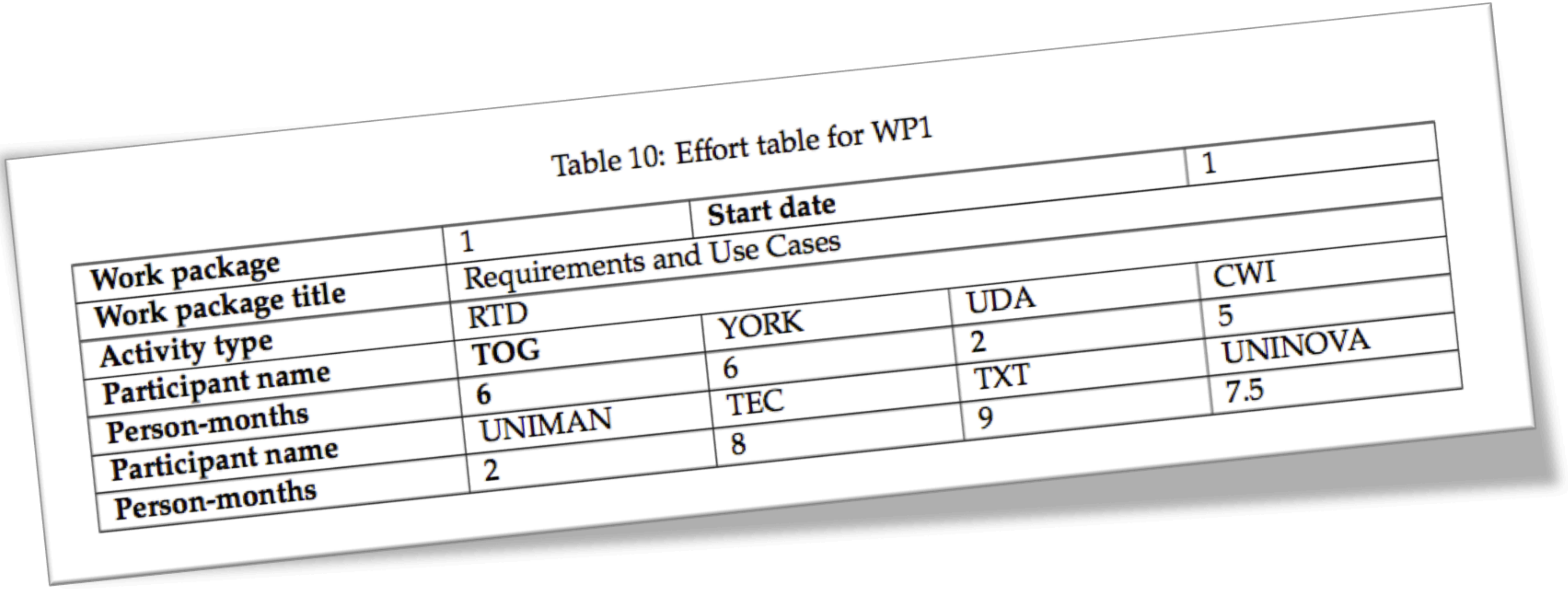

Table 10: Effort table for WP1

Work package	1	**Start date**		1
Work package title	Requirements and Use Cases			
Activity type	RTD			
Participant name	**TOG**	YORK	UDA	CWI
Person-months	**6**	6	2	5
Participant name	UNIMAN	TEC	TXT	UNINOVA
Person-months	2	8	9	7.5

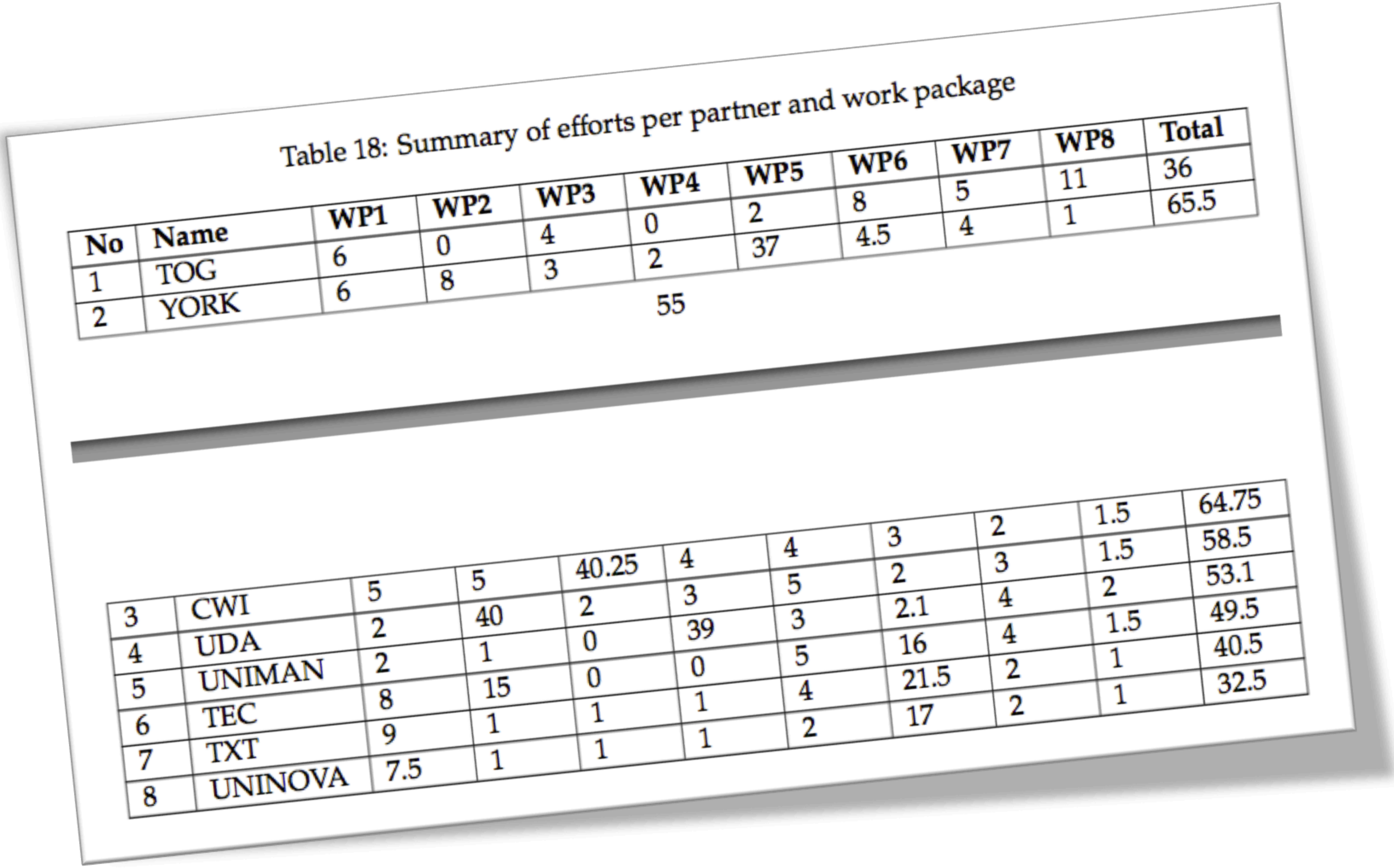

Table 18: Summary of efforts per partner and work package

No	Name	WP1	WP2	WP3	WP4	WP5	WP6	WP7	WP8	Total
1	TOG	6	0	4	0	2	8	5	11	36
2	YORK	6	8	3	2	37	4.5	4	1	65.5
3	CWI	5	5	40.25	4	4	3	2	1.5	64.75
4	UDA	2	40	2	3	5	2	3	1.5	58.5
5	UNIMAN	2	1	0	39	3	2.1	4	2	53.1
6	TEC	8	15	0	0	5	16	4	1.5	49.5
7	TXT	9	1	1	1	4	21.5	2	1	40.5
8	UNINOVA	7.5	1	1	1	2	17	2	1	32.5

55

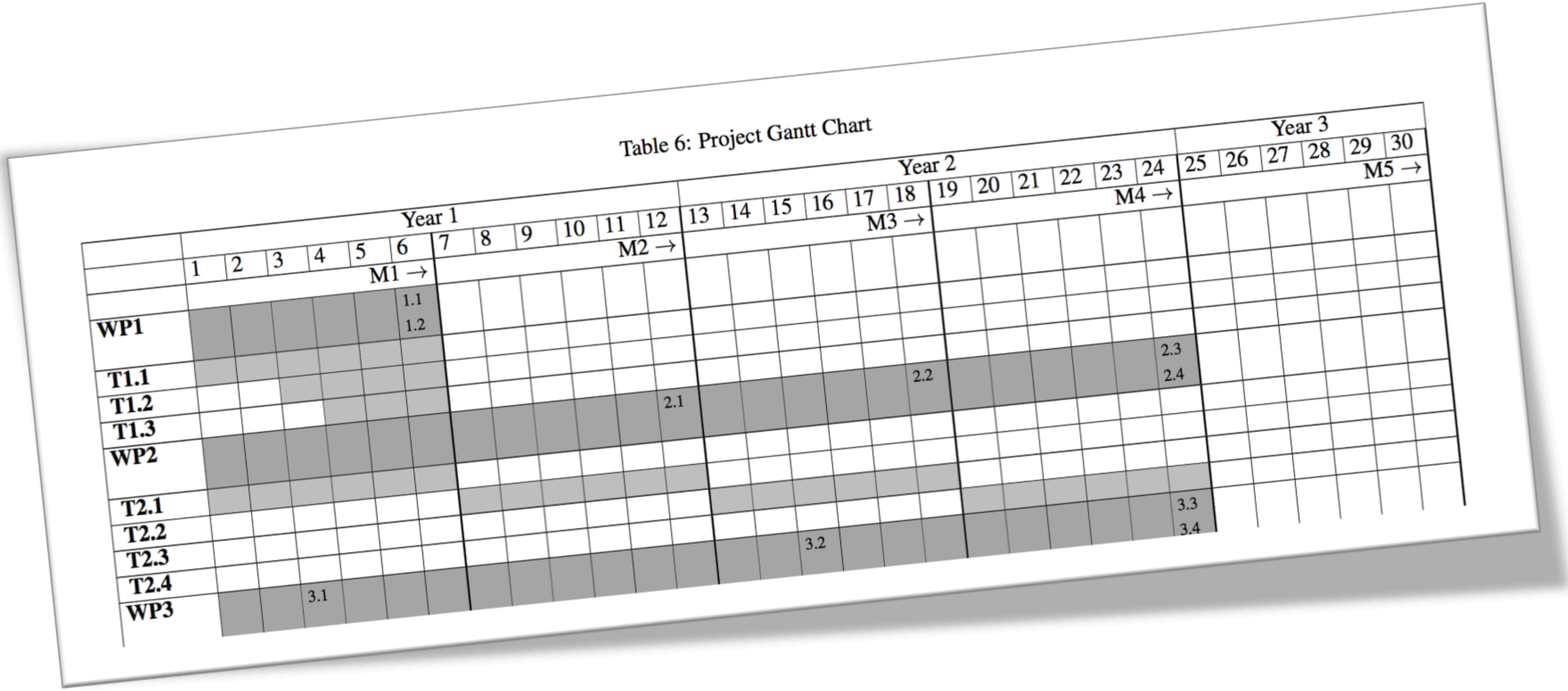

Table 6: Project Gantt Chart

	Year 1												Year 2												Year 3					
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
						M1 →						M2 →						M3 →						M4 →						M5 →
WP1	■	■	■	■	■	■ 1.1 1.2																								
T1.1	▒	▒	▒	▒	▒	▒																								
T1.2			▒	▒	▒	▒																								
T1.3				▒	▒	▒																								
WP2	■	■	■	■	■	■	■	■	■	■	■	■ 2.1	■	■	■	■	■	■ 2.2	■	■	■	■	■	■ 2.3 2.4						
T2.1	▒	▒	▒	▒	▒	▒																								
T2.2							▒	▒	▒	▒	▒	▒																		
T2.3													▒	▒	▒	▒	▒	▒												
T2.4																			▒	▒	▒	▒	▒	▒						
WP3	■	■	■	■ 3.1	■	■	■	■	■	■	■	■	■	■	■	■ 3.2	■	■	■	■	■	■	■	■ 3.3 3.4						

What to do?

- Create a DSL for designing such projects (that is, specify a metamodel using the UML class diagram syntax).
- Create a model that conforms to the DSL and exercises all its features at least once
  - Use the UML object diagram syntax for this.