Deliverable number:	D2.3
Deliverable Title:	Background data structure – Update of D2.1
Type (Internal, Restricted, Public):	PU
Authors:	H. Langer, D. Nyga, M. Beetz, I. Markievicz, D. Vitkute-Adzgauskiene, M. Tamosiunaite, F. Wörgötter
Contributing Partners:	UoB, UGOE, VDU

Project acronym:	ACAT
Project Type:	STREP
Project Title:	Learning and Execution of Action Categories
Contract Number:	600578
Starting Date:	01-03-2012
Ending Date:	30-04-2015

Contractual Date of Delivery to the EC:	30-11-2015
Actual Date of Delivery to the EC:	30-11-2015

Content

1. Executive summary

This deliverable according to the DOW is a short update of the deliverable D2.1. Here we present new details introduced into the ACAT textual ontology and the Action Data Tables (ADT) and we indicate how we are making a link between ADTs and Probabilistic Action Cores (PRACs).

The main update for the textual action ontology is by introducing of the WordNet-style hierarchical structure, which allows using the ACAT ontology together with WordNet. In the ADTs we have changed quite some fields since D2.1, but most of the changes have been introduced already in PPR2. Here we indicate the new structure and emphasize those fields which were added after PPR2. The PRACs are linked to ADTs through a unique resource identifier (URI) link and can be used together with the ADT material when a PRAC exists.

2. Introduction

This deliverable is to provide an update on the data structures introduced in D2.1. The main ACAT data structure consists of a language-based action ontology (symbolic knowledge) as well as the Action Data Tables (ADT) where robot experience is stored up to signal level detail. ADTs are attached to the symbolic ontology and provide a symbolic to sub-symbolic link. Additional data structures produced by the ACAT project are the Probabilistic Action Cores (PRACs) which store information about specific high level actions, like “adding” or “neutralizing”. The objective of the deliverable is to explain how language ontology and ADTS have been adapted based on the project needs (as compared to the version presented in D2.1) as well as how the PRAC data structures are integrated into the ADTs.

3. ACAT data structure updates

In this chapter we will show how the ACAT data structures were changed:

- By introducing new relations and properties into the ACAT language-based ontology;
- By introducing new entities into the Action Data Tables (ADTs);
- By linking Probabilistic Action Cores (PRAC) to ADTs.

3.1. New relations and properties in the action ontology

The ACAT action ontology was organized in a hierarchical manner based on the WordNet structure and supplemented with the following information from Wordnet:

- *Synonym words* for each action word, organized into synsets.
- *Hypernym words* for each action word. Hypernym is a linguistic term for a word whose meaning includes the meanings of other words.
- *Hyponym words* for each action word where available. A hyponym word is a word whose semantic field is included within that of another word.
- *Troponym words* for each action word where available. Troponym is a verb that indicates more precisely the manner of doing something by replacing a verb of a more generalized meaning.

The introduced hypernyms, hyponyms and troponyms allow addressing bigger number of natural language instructions, which are expressed not in the same (as robotic system has been performing before), but WordNet-related terms.

We have added properties to actions in the ontology: ”main action”, ”robotic action” and ”supportive action”. Definitions of the properties are (taken from the ACAT term glossary presented in PPR2):

- **Main action (word):** the *action word* that identifies the *ADT-action* of the attached *ADT* uniquely.
- **Supportive action (word):** is the *action word* that can be told or omitted when describing the main action (by means of *main action word*) in instructions, e.g. *supportive action words* "locate", "grasp", "pick up". E.g. “*Pick* the bottle and place it onto the counter.”
- **Robotic action (word):** a word that indicates an action which can be directly and straightforwardly described by an *action class* and/or *action primitive* spelled out in the ACAT project. The initial set of *robotic action words* is created by a human labeling, but the set is increased automatically through synonymy analysis.
- **Non-robotic action (word):** an a*ction word* that cannot be described within the *action classes* or *action primitives* spelled out in the ACAT project.

These added action properties are required in interpreting natural language instructions by the symbolic compiler.

3.2. New entities in the Actions Data Tables (ADTs)

The ADT structure is described in Table 1 below. Most of the fields correspond to the ADT structure provided in PPR2, section ”Detailed report – WP2”. However, a few new fields are added. These are: ”Action links”, which allows linking ADTs to external data structures. Currently we are interested in linking with Probabilistic Action Core (PRAC) structure. The other new component is ”Movement primitives”,

which adds another hierarchical level to the ADT (like ”move arm” or ”grasp”). Each action chunk can have one or more movement primitives. Adding movement primitives allows re-execution of the ADT by a wide class of robotic systems without the need to look into the attached rosbag. The newly added fields are marked using green background.

Table 1. Updated ADT structure

Action Data Table (ADT)		
HEADER		
Name	**Description**	**Type**
Instruction	*Textual instruction as given in instruction sheet*	*Text*
Action	Action type (name, same as above, verb)	string
Action links	A pointer to a Probabilistic Action Core (PRAC) data structure	URI pointer
Main object	*Object name (noun)*	*String*
Main Object Descriptors: 1) CAD model 2) pose *3) part constitution* *4) material* *5) size* *6) mass*	Robotics relevant object description: 1) 3D model (CAD /pointcloud) or a list of models (alternatively, image or a list of images) 2) Pose *3) Part graph and part models* *4) symbolic notation of material object is made of* *5) Approximate size in meters* 6) *Approx mass in kg.*	1) pointers array to CAD and image library 2) 7 real numbers for pose *3) pointers to files* *4) text* *5) 3 real numbers* 6) *real number*
Primary object	*Object name (noun)*	*string*
Primary Object Descriptors: same as "Main object descriptors"		
Secondary object	*Object name (noun)*	*string*
Secondary Object Descriptors: same as "Main object descriptors"		
Tool	*Object name (noun)*	*string*
ToolDescriptors: same as "Main object descriptors"		
Main support plane	*Object name (noun)*	*String*
Plane descriptors	*Pose*	*7 real numbers*
Primary support plane	*Object name (noun)*	*String*
Plane descriptors	*Pose*	*7 real numbers*
Secondary support plane	*Object name (noun)*	*String*
Plane descriptors	*Pose*	*7 real numbers*
Tool support plane	*Object name (noun)*	*String*
Plane descriptors	*Pose*	*7 real numbers*
Wrist-to-TCP-transform	*Pose*	*7 real numbers*
Recording method	*Which way robot data was obtained, e.g.," kinesthetic guidance"*	*String*
Recorded data	*Trajectories, forces, etc. recorded in the rosbag file*	*pointer to a file*
Anchor Points (SEC)	Semantic Event Chain for this action which **defines the chunks of an action**	Array of strings (for interpretations)
Action primitive sequence *1) name* *2) descriptors* *3) name* *4) descriptors* *...*	*1), 3), ...: Name(verb) for each action primitive* *2), 4), ...: start and end chunk for each action primitive*	*1), 3),..:. text* *2), 4), ...: two integers for each action primitive*
SEQUENCE OF ACTION CHUNKS		
SEC chunk 1		
Wrist or TCP characteristics: *1) start time* *2) start pose*	*Following characteristics are shown:* *1) Start time of the action chunk as in rosbag* *2) Pose of the robot wrist/TCP at start time* *3) Force of the robot manipulator at start time*	*1) 1 real number* *2) 7 real numbers* *3) 6 real numbers*

3) start force *4) end time* *5) end pose* *6) end force*	*4) End time of the action chunk as in rosbag* *5) Pose of the robot wrist/TCP at end time* *6) Force on the robot manipulator at endt time*	*4) 1 real number* *5) 7 real numbers* *6) 6 real numbers*
Main object characteristics: *1) start time* *2) start pose* *3) end time* *4) end pose*	*Following characteristics are shown:* *1) Start time of the action chunk as in rosbag* *2) Pose of the robot wrist/TCP at start time* *3) End time of the action chunk as in rosbag* *4) Pose of the robot wrist/TCP at end time*	*1) 1 real number* *2) 7 real numbers* *3) 1 real number* *4) 7 real numbers*
Primary object characteristics: *1) start time* *2) start pose* *3) end time* *4) end pose*	*Following characteristics are shown:* *1) Start time of the action chunk as in rosbag* *2) Pose of the robot wrist/TCP at start time* *3) End time of the action chunk as in rosbag* *4) Pose of the robot wrist/TCP at end time*	*1) 1 real number* *2) 7 real numbers* *3) 1 real number* *4) 7 real numbers*
Secondary object characteristics: *1) start time* *2) start pose* *3) end time* *4) end pose*	*Following characteristics are shown:* *1) Start time of the action chunk as in rosbag* *2) Pose of the robot wrist/TCP at start time* *3) End time of the action chunk as in rosbag* *4) Pose of the robot wrist/TCP at end time*	*1) 1 real number* *2) 7 real numbers* *3) 1 real number* *4) 7 real numbers*
Tool characteristics: *1) start time* *2) start pose* *3) end time* *4) end pose*	*Following characteristics are shown:* *1) Start time of the action chunk as in rosbag* *2) Pose of the robot wrist/TCP at start time* *3) End time of the action chunk as in rosbag* *4) Pose of the robot wrist/TCP at end time*	*1) 1 real number* *2) 7 real numbers* *3) 1 real number* *4) 7 real numbers*
Grasp characteristics: *1) grasp status* *2) grasp type* *3) grasp pose* *4) grasp force* *5) success*	*1) Indicates if in the action chunk robot needs to grasp, to ungrasp or make no change to the grasp status;* *2) grasp type from a grasp type table* *3) wrist/TCP pose* *4) normalized force for gripper closing* *5) success of a grasp*	*1) string* *2) string* *3) 7 real numbers* *4) number from interval [0, 1]* *5) true or false*
Movement primitives: *1) movement primitive sequence* *2) start time for each movement primitive* *3) end time for each movement primitive* *4) parameters for each primitive*	*1) Indicates the sequence of movement primitives in the chunks, eg. 1."hand preshape", 2."move arm", 3."grasp".* *2) Start time of each movement primitive in a rosbag* *3) End time of each movement primitive in a rosbag* *4) Different primitives have different parameters, e.g. for "move arm" primitive parameters are start and end pose of the TCP, while for "grasp" parameter is the grasp force, etc.*	*1) sequence of strings* *2)1 real number* *3) 1 real number* *4) numerical values, depending on the primitive type*
Action chunk success specifier: *1) SEC transition* *2) trajectory* *3) force* *4) pose* *5) failure description*	*Overall success to reach the desired end state* *1) was the desired SEC transition achieved?* *2) did the trajectories match to planned ones?* *3) did the forces match the planned ones?* *4) were the required end poses achieved?* *5) textual description of failure*	*1) true or false* *2) real number (1:full success)* *3) real number (1: full success)* *4) real number (1: full success)* *5) text*
SEC chunk 2		
Same as for SEC chunk1		
More chunks if needed		
Overall action success specifier: *1) precondition success* *2) overall success* *3) failure anchor*	*1) Were preconditions fulfilled by a scene?* *2) Was the action overall succesful* *3) If not, which chunk has failed*	*1) true or false* *2) true or false* *3) integer number*

3.3. Linking Probabilistic Action Cores to ADTs

Here we discuss the link between Probabilistic Action Cores (PRACs) and ADTs. PRAC is a learning and reasoning framework of action-specific first-order probabilistic knowledge bases that are able to interpret

instructions formulated in NL and can be used to infer the most probable completion of an action with respect to its abstract, symbolic parameterization. (Probabilistic Action Core is a unit of PRAC framework.)

As an example, consider the natural-language instruction “neutralize 75 ml of hydrochloric acid” from the CHEMLAB scenario. In this example, Neutralize (in a chemical sense) represents an action, which has attached two parameters, namely an AcidSubstance and an AlkalineSubstance, which both must be known in order to perform the neutralization. However, in the original instruction, the alkali counterpart is not specified but needs to be inferred. A graphical representation of such an inference task is shown in Fig. 1.

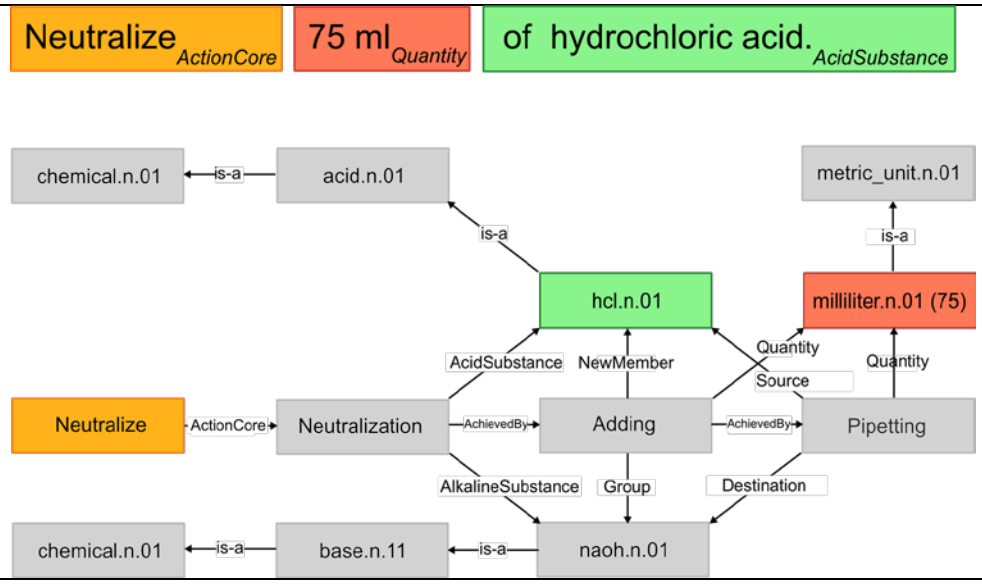

Fig. 1. Graphic representation of instruction interpretation using PRAC.

ADTs consider a small number of roles for symbolic arguments given in the instruction. Only main, primary, secondary objects and tool, as well as corresponding support planes, are indicated (see Table 1). For performing reasoning at a higher level this is insufficient. E.g. for the Probabilistic Action Core “neutralize”, indicated in the figure above, we need the roles of “AcidSubstance” and “AlkalineSubstance”. The same problems arises when considering an 'adding' or 'pipetting' action. In order to parameterize such actions appropriately for execution, not only main, primary and secondary objects, but also an amount of the added substance and a measurement unit are needed, such as amount (75) and unit (milliliter.n.01). Since the actions considered in the demonstration scenarios IASSES and CHEMLAB are very versatile, for higher level reasoning it makes sense to attach to different actions different sets of roles that are specific to these actions, since the mentioned fixed set of roles might be insufficient for some actions. Thus, we add a link to an ADT that indicates this connection by means of a unique resource identifier (URI) to the appropriate Probabilistic Action Core, when such cores are existing (See line with green background close to the top of Table 1). A list of actions for which Probabilistic Action Cores (PRACs) exist with appropriate roles shown in Table 2 below.

Table 2. A list of existing Probabilistic Action Cores with action-specific roles

Actioncore (Senses and Roles)	**#Training Sets**	**Roles**	**Examples**
Adding	35	**Theme**: Object to be added. **Goal**: Object to contain the theme object. **Action Verb**: Verb which triggers Actioncore. **Amount**: Amount of the theme defined as number. **Unit**: Unit of amount. For instance gram or milliliter.	Start with adding 1 liter of water to the chlorous acid.
Filling	11	**Stuff**: Object to be filled into the goal object. **Goal**: Object to store the stuff object. **Action Verb**: See Adding AC.	Fill a mixer with pineapples.
Flavouring	9	**Spice**: Spice used to flavour the goal object. **Goal**: Object to be flavoured. **Action Verb**: See Adding AC.	Flavour the tomato sauce with the oregano.
Neutralizing	10	**Neutralizee**: Object to be neutralized. **Neutralizer**: Object used to neutralize the neutralizee. **Action Verb**: See Adding AC. **Amount**: Amount of the neutralizer object defined as number. **Unit**: Unit of the amount. For instance milliliter.	Neutralize the methacrylic acid with 100 milliliters of cyanuramide.
Opening	3	**Obj To Be Opened**: Object which should be opened. **Action Verb**: See Adding AC.	Open the drawer.
Pouring	10	See Filling AC.	Pour some water into the cup.
Preheating	3	**Obj To Be Heated**: Object which should be heated. **Temperature Unit**: The unit of the temperature setting. For instance Celsius. **Temperature Setting**: A number defining the temperature. **Action Verb**: See Adding AC.	Preheat the oven to 100 degree Fahrenheit.
Starting	1	**Obj To Be Started**: Object which should be turned on. **Action Verb**: See Adding AC.	Start the centrifuge.

Actioncore (Achieved By)	**#Training Sets**	**Roles**	**Examples**
Unscrewing	1	**Obj To Be Unscrewed**: Object which should be opened with unscrewing. **Action Verb**: See Adding AC.	Open the test tube.
Pulling	1	**Obj To Be Pulled**: Object which should be opened with pulling. **Action Verb**: See Adding AC.	Open the drawer.
OpeningADoor	2	**Obj To Be Opened**: Object with a door which can be opened. **Action Verb**: See Adding AC.	Open the fridge.
OperatingATap	6	**Liquid**: Liquid to be filled in the goal object. **Goal**: Object to store the liquid. **Action Verb**: See Adding AC. **Amount**: Amount of the liquid defined as number. **Unit**: Unit of the amount. For instance milliliter.	Fill a cup with water.
UsingSpiceJar	10	**Content**: Object contained in the measuring cup. **Goal**: Object which should be flavoured. **Action Verb**: See Adding AC.	Flavour the chicken with pepper.
Spooning	10	**Substance**: Object contained in the spoon. **Goal**: Object to contain the substance. **Action Verb**: See Adding AC.	Add some bananas to the mixing bowl.
TurningOnElectricalDevice	4	**Device**: Device to be turned on. **Action Verb**: See Adding AC.	Preheat the oven.
UsingMeasuringCup	6	**Content**: Object contained in the measuring cup. **Goal**: Object to contain the content of the measuring cup. **Action Verb**: See Adding AC. **Amount**: Amount of the content defined as number. **Unit**: Unit of amount. For instance milliliter.	Start with adding 1 liter of water to the chlorous acid.
Pipetting	6	**Content**: Object contained in the pipette. **Goal**: Object to contain the content of the pipette. **Action Verb**: See Adding AC. **Amount**: Amount of the content defined as number. **Unit**: Unit of amount. For instance drops.	Start with neutralizing the pyridine with 4 drops of the hydrofluoric acid.

4. Conclusions

This deliverable has introduced updates to the most important parts of the ACAT data structure (consisting of the textual language ontology and the Action Data Tables ADTs), as well as updates on the connection between ADTs and Probabilistic Action Cores (PRAC).

These structures will be used in the final demonstrators of the ACAT project, as well as benchmarking procedures. Thus, evaluations of the utility of those structures will be available at the end of the project.