R5-COP

Reconfigurable ROS-based Resilient Reasoning Robotic Cooperating Systems

Knowledge and Database

Tadeusz Dobrowiecki (BME), Itziar Landa Torres, Jesus Lopez Lobo (TRI), Jon Azpiazu (SIN)

Project	R5-COP	**Grant agreement no.**	621447
Deliverable	D26.22	**Date**	10/31/2016
Contact Person	Tadeusz Dobrowiecki	**Organisation**	BME
E-Mail	dobrowiecki@mit.bme.hu	**Diss. Level**	R/PU

Document History			
Ver.	**Date**	**Changes**	**Author**
0.0	6.10.2016	Lay-out of the deliverable	Tadeusz Dobrowiecki (BME)
0.1	15.10.2016	Database and record explanations	Itziar Landa Torres, Jesus Lopez Lobo (TRI)
0.2	17.10.2016	Review of knowledge and architecture issues	Tadeusz Dobrowiecki (BME)
0.3	18.10.2016	Comments to database	Jon Azpiazu (SIN)
0.4	23.10.2016	General Comments	Itziar Landa Torres (TRI)
0.5	24.10.2016	Comments to database	Jesus Lopez Lobo (TRI)
0.6	28.10.2016	Comments to database	Jesus Lopez Lobo (TRI), Jon Azpiazu (SIN)
0.7	30.10.2016	Review comments	Thomas Almdal (DTI) György Strausz (BME)

Note: Filename should be

"R5-COP_D##_#.doc", e.g. „R5-COP_D91.1_v0.1_TUBS.doc"

Fields are defined as follow

1. Deliverable number ***.***

2. Revision number:

draft version **v**

approved **a**

version sequence (two digits) ***.***

3. Company identification (Partner acronym) *****

Content

List of Acronyms

CFR	Co-worker robots
CWA	Closed World Assumption
DB	Data Base
GA	Genetic Algorithm
IT	Information Technology
JSON	Java Script Object Notation
KB	Knowledge Base
KDB	Knowledge Database
KML	Keyhole Markup Language
PRA	Planning and Reasoning Architecture
RK-MoHS	Random Keys based Multi-objective Harmony Search
ROS	Robot Operating System
TSP	Travelling Salesman Problem
WM	Working Memory

1 Introduction

1.1 Summary

We present the final view of the Knowledge and Database (KDB) component for the co-worker robot performing autonomous inspections in a gas-and-oil facility. After a short review of the knowledge content and architecture issues treated in the earlier deliverables D41.20 “Integration of configurable components and application development”, D26.12 “Planning and Reasoning Architecture”, D26.21 “Knowledge and Database (tentative)”, and D26.30 “Task Planning and Representation”, a detailed review of the database records implementing the KDB is given, synchronized with the actual developments with the co-worker prototype at SINTEF. We refer also to D26.40 „Plan execution and error handling” for further details related to the Knowledge and Database management.

1.2 Purpose of document

Our aim is to show how the preliminary ideas about the knowledge demands of the co-worker use case and the preliminary concepts about the architecture and the coverage of its knowledge intensive components were tuned to the developing requirements of the actual co-worker prototype to produce an integrated Knowledge and Database component (KDB) of the co-worker architecture (in principle of its Planning and Reasoning Architecture, but the conceived final KDB provides services to other system components also).

1.3 Partners involved

Partners and Contribution		
Short Name	**Full Name**	**Contribution**
BME	Budapest University of Technology and Economics (HU)	Preparation of the report, internal review
TRI	Fundaction Tecnalia Research & Innovation (ES)	Preparation of the report
SIN	Stiftelsen SINTEF (NO)	Preparation of the report
DTI BME	Danish Technological Institute (DK) Budapest University of Technology and Economics (HU)	Internal review

2 Pilot use case and its knowledge management

The Knowledge and Database component (KDB) described in the followings is the final version of the Knowledge and Database component for an industrial co-worker robot performing inspections in the open-air large industrial oil and gas production facility, reported in the series of the WP41 deliverables (especially in D41.20 “Integration of configurable components and application development”) (this robot will be designated occasionally in the followings as the CFR). In such facilities field operators are required to perform regular inspections and maintain equipment, being frequently exposed to extreme climate conditions and even to high levels of toxic and corrosive gases, fumes, or vapours [5, 11, 12, 17]. It is thus an imperative to free as much of the human factor as possible from this burden and it is also advantageous to move expensive sensors around on a mobile platform, instead of covering the facility with a permanent sensor network [3, 4, 7, 9, 10, 16].

The KDB component was introduced into the CFR system architecture in the D26.12 “Planning and Reasoning Architecture”, and described separately in the D26.21 “Knowledge and Database (tentative)”. The preliminary implementation as an integrated database component was presented in the D26.30 “Task Planning and Representation”.

2.1 Short introduction to the industrial co-worker field

In the basic (inspection) version of the use case the CFR robot is expected to purposefully wander (after the thorough planning and scheduling of its movements) along the facility, visiting particular equipment or installation hot-points, measuring there essential production, safety or pollution parameters, and sending inspection information and sensor data to a human operator located remotely. Furthermore the CFR is expected to keep continuously an eye and reporting on abnormal values and events. Upon evaluating the received information, at any point in the process, the operator can decide to take control over the robot and teleoperate it, controlling the inspection by himself. Under emergency the robot will behave as a "first responder", reaching rapidly the origin of the alarm and helping the operator to asses the situation [10].

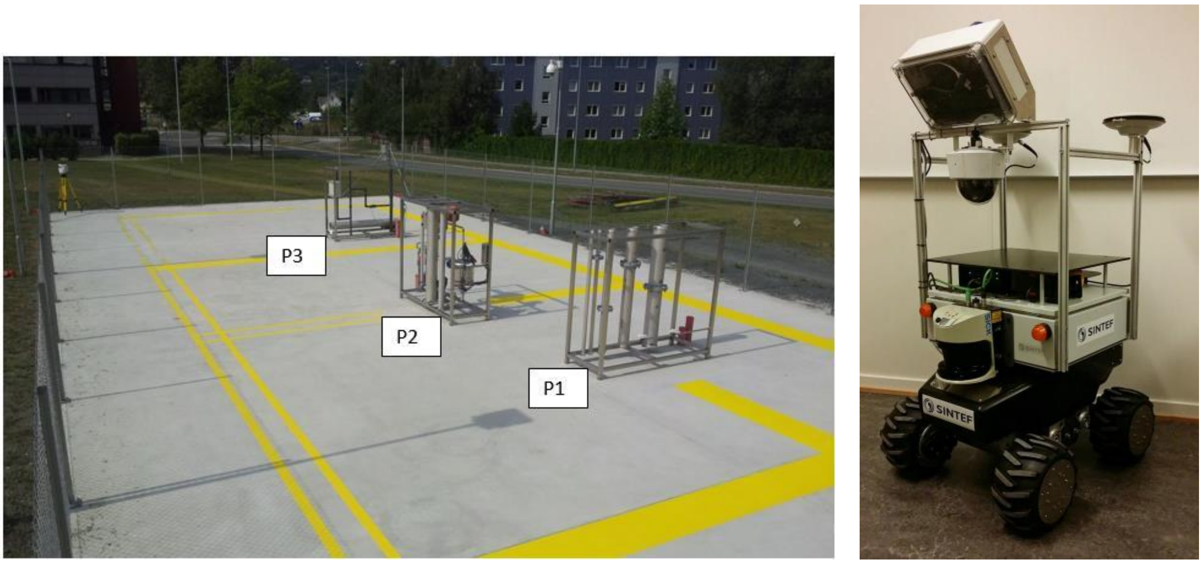

Fig. 1. Trondheim-based open-air lab facility and the CFR robot prototype (from D41.20).

The co-worker prototype is tested in an outdoor laboratory area located in Trondheim, which emulates some of the common elements that can be found in most oil and gas platforms. The northerly location makes also feasible the testing of different weather conditions.

The co-worker must possess sensors to avoid collisions with another equipments and personnel, to calculate and report the position of the mobile base, to assess and inspect the events being responded to, and also to give the operator an overview of the situation in the real-time. Its body of actions should cover movements, maneuverability, making (and interpreting) observations, and communicating with the operator (and possibly with other agents).

The principal goal is to timely (and possibly autonomously) inspect the whole installation and to provide the operator with the best situation awareness possible. This must be done being no slower than a human in the same task, not inferring with the work of human specialists on the same installations, to avoid moving obstacles (pedestrians, cyclists, motorists, animals(?)) in a "human" way, to decide how to access different equipment and to judge which information is important to report, and all this with an eye kept upon the self-preservation.

An inspection mission is composed by a sequence of operations that the robot must execute. Typical operations consist of navigating to a waypoint, acquiring sensor data and requesting feedback from the operator. The CFR must also be able to react to some unpredictable situations that might alter the execution of the plan (like encountering blocked path, loosing connection with control centre, spending battery, facing sensor failure, running into unanticipated harsh weather conditions, encountering abnormal situation, confronting people on the road, or yielding to the operator controls (assisted teleoperation)).

The essential conclusion is that the adequate task execution and the quality of the co-worker actions depends on a very large body of heterogeneous knowledge and data, ranging from simple facts to 2D maps, describing the structures in the environment, dynamic models of moving obstacles, and paralleled by reasoning and learning, coping effectively with local/global, static/dynamic, and certain/uncertain issues.

2.2 CFR knowledge

For the better understanding of the knowledge intensive issues we can make a distinction between the domain, the application, and the case knowledge (and data). **Domain knowledge** encompasses the environment and the agency of the robotic system. It presents the model of the task environment and lists the required sensory and actuator modalities to make this environment accessible (at least in part) to the robot. The domain knowledge is decisive in designing the architecture (hardware and software) of the robotic system and in choosing the algorithms for the higher level system functions and modules.

In the CFR case the knowledge about the task environment may extend to plant lay-out maps, access road maps, inspection points, their access ways, types of equipment, specified measurement protocols at inspection points, computable characteristics from direct measurements (sensor fusion), inspection point dependent observations pertained to abnormal states and disasters, evacuation roads, locations of other services (repair, charging, first-aid, ...), kinds and models of moving obstacles (pedestrians, cyclists, motorists, vehicles), weather patterns, kinds of local meteorological measurements (properties of wind, perspiration, storms, ...).

Element	Description
V	Power supply
C	Pipes
VB	Ball valve
VG	Globe valve
T	Tank
G	Dial gauge
W	Water tank
	Electrical terminal

Fig. 2. Types of inspection points to be found in the test facility (from D41.20).

The agency of the CFR demands a wide spectrum of sensory and actuator equipment. We can distinguish between sensor modalities on the mobile base to avoid collisions with equipment and personnel, to calculate and report the position of the mobile base, payload sensors to assess the type and severity of an abnormal situation, giving the operator an overview of the situation in real-time (vision, hearing), and assessing the acting fitness of the robot.

The acting skills of the co-worker are walking, drawing near the inspection points with due attention to the safety and quality of observations, self-diagnosing, observing equipment accordingly to the observations receipts, communicating (reporting protocols) with the operator, avoiding the potential obstacles, issuing warnings.

The detailed description of the co-worker sensory and actuator hardware is presented in the deliverable D41.20.

The **application knowledge** is the body of knowledge characterizing tasks demanded from the robot and realizable given the domain knowledge. Such knowledge is expressed usually as various goals refined by requirements and posed limitations and in a suitably coded form this knowledge is also a subject to the reasoning (usually planning) algorithms.

In the case of the CFR the single global goal is to provide the installation operator sufficient and timely situational awareness in order to assess the situation and make informed decision, especially in adverse external conditions and in case of abnormal and disastrous events.

As a by-product we should also aim at reducing the logistic and manpower costs associated with worker-performed inspections, reducing the number of fixed sensors at the site and the cost and risk associated with workers installing and maintaining these sensors, minimizing the number of human interventions due to (miscalculated) late charging, robot provoked traffic accidents, stoppages in travel, access, and measurements, minimizing the number of false positives (false alarms) and false negatives (unreported problems) in the inspection reports.

The primary requirements can be stated as solving the interaction between the operator and the co-worker in such a way not to encumber the operator with robot navigation and control issues, but purveying the operator with the right con-text and freedom to investigate. This paralleled by the ability to deploy rapidly, to avoid collisions with equipment and personnel in a "human-like" manner, to be "survival conscious" and to avoid being damaged while performing its duties, and to be adaptable to varying environments (within the domain model).

The **case knowledge and data** is related to the fact that the co-worker collects and processes a large volume of data during the inspection run. Under normal operating conditions (for the plant and the co-worker) the majority of these data will be discarded after being used to make control decisions. In case of abnormal functioning or emergency however more data must be stored for later post-mortem evaluation, as training data for human and co-worker learning.

Here we can distinguish the actual weather patterns, actual local meteorological measurements, probability and intensity of weather related events (snowing, raining, storms, ...), shift inspection plans optimized for actual situation, final executed (modified) inspection plans and schedules, summary reports for the shift, measurements and events experienced at particular inspection points, description of the travelling conditions, traffic data, expected and unexpected obstacles, description of events (interrupting events, blocking situations, conflicts, events not pertained to the inspection, nevertheless observed by the robot, ...), finally traces of the CFR-operator communication and operator's decisions (permanently documented and stored).

2.3 Knowledge management

Part of the knowledge pertinent to the co-worker use case must be evaluated by the system designer and finds its way into the co-worker system architecture and sensory-actuator design at the hardware and software level. A part of the knowledge is however the “working knowledge” of the co-worker robot, used to pre-compute action plans, to adapt them to the actual circumstances, and to interpret the results of the plan execution. Such knowledge (and data) must be stored in a component accessible in the run-time by the body of the co-worker algorithms.

The notion of knowledge base (KB) was introduced traditionally in the early expert systems, built as an architecture comprising Knowledge Base (KB), Working Memory (WM), and Inference Engine. KB was usually designed to support momentary inferences, but also to be extended, updated, checked for validity, mistakes, also to be read by humans, to evaluate and to extend their professional knowledge.

In robotic intelligent systems a different KB module architecture is needed. The capacity of the system intelligence is decided by the application requirements and provided by design. This capacity is usually not to be extended in the lifetime of the system, as a robotic system only partially able to provide its services, and learning, extending its knowledge over time may be interesting, even useful in some applications, nevertheless it is not the mainstream of intelligent robotic applications. Intelligent robotic systems may be confronted with new elements of the environment, objects, situation, but it is rather new "data", than a new "knowledge".

There exists of course already a new generation of robotic intelligence, which can learn by observation, communicate with humans in natural language, and most of all important, may not only accept the factual knowledge about new objects and how to handle them, but may predict their affordances, i.e. possible and unanticipated usages, even to the operator presenting them to the robot, extending thus considerably the spectrum of the services the robot could provide [1, 2, 15].

In the co-worker context, however, in demanding industrial environment and well defined professional activity there is no place for intelligent robot which gradually improves and extends its knowledge faculties and maintains a KB designed to this purpose. The intelligence of the robot should be designed well from the very beginning and should vary little during its service time. If some gross changes would be required affecting also the KB, it should belong to the testing phase of the prototype design.

As a consequence the need to learn new things is limited rather to account for smaller differences in the environment, using these data to tune algorithms rather than to design new acting opportunities, not to speak about the affordances.

There are three principal activities based on the content of the system knowledge base: task planning, mission control with re-planning, and learning/reasoning.

The **task planning and re-planning** is described schematically in relation with the Planning and Reasoning Architecture in D26.12, and at the applied algorithm level (task selector: Random Keys based Multi-objective Harmony Search (RK-MoHS), router: a TSP algorithm based on GA and scheduler) in detail in D26.30. It resides in the Mission Control module of the architecture (see Fig. 3) and uses domain and application knowledge to derive shift dependent routes and activity schedules (i.e. case level knowledge).

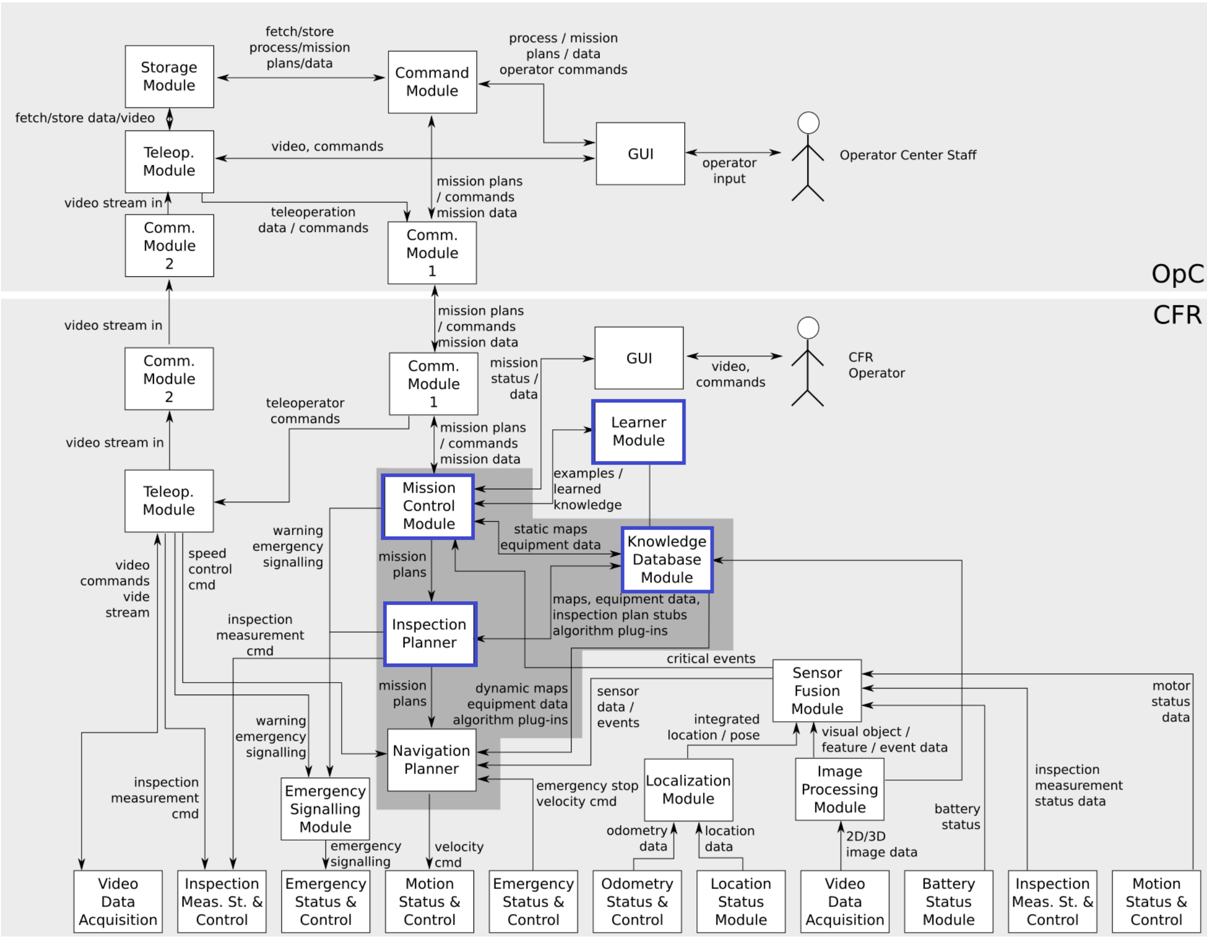

Fig. 3 Architecture of the CFR (D26.12). Shaded area is the Planning and Reasoning Architecture and the modules highlighted in blue are the principal clients of the Knowledge and Database component.

Upon execution of the planned activities, at any point in the process, some unexpected events can arise (termed triggers and described in more detail in D41.20 and D26.30), and then special rutines are applied (**re-planning**). They oblige the CFR to adapt to the new circumstances, such as bad weather and poor visibility conditions, sudden alerts, abnormal measurements, communication failures, or any kind of problems that makes it impossible for the CFR to continue inspecting. The re-planning usually means re-selection, i.e. reducing (e.g. due to weather, visibility) or increasing (e.g. due to abnormal measurements) the number of tasks (inspected points) in the mission, and then arranging new optimal routes to conclude the inspection.

The option of **learning and reasoning** was introduced in D26.30 and is described in the algorithmic detail in D26.40. The CFR collects plenty of various data, which – especially when unanticipated events occured and there was a need for re-planning – may contain valuable implicit information helpful to avoid such situations in the future (by e.g. improving on the planning, or on the reactive procedures in the run-time). Before the introduction of the robotic co-worker such information was collected in human readable logs and reports, passed higher in the facility hierarchy and analyzed there to improve on the procedures and regulations. The availability of such data electronically paired with the presence of the IT platform able to run machine learning algorithms opens the opportunity to automatize also this aspect of the industrial task. From various forms of learning the supervised inductive learning seems the most natural choice (as opposite to the reinforcement learning), as the qualification (labelling) of the training examples is ready at hand. To make the later learning feasible the collected data must be thus already encoded into training examples containing information about the battery level vs. plan, weather conditions, abnormal and critical measurements, path dependencies, already activated triggers, and dependency on the direct operator control.

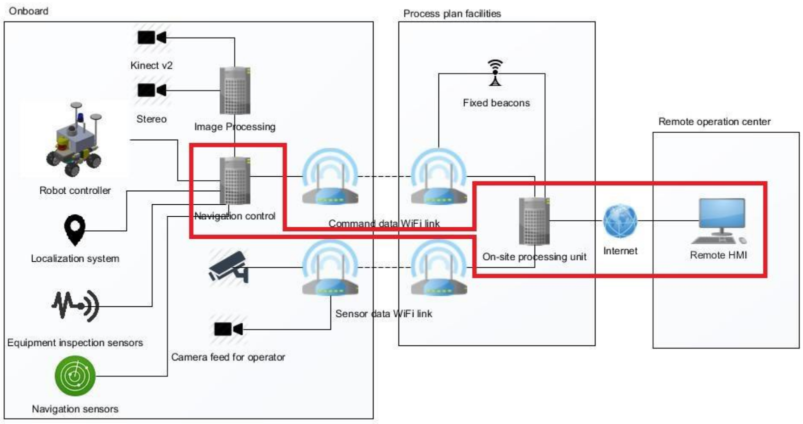

Fig. 4. The integrated information system of the industrial facility (D41.20, D26.12).

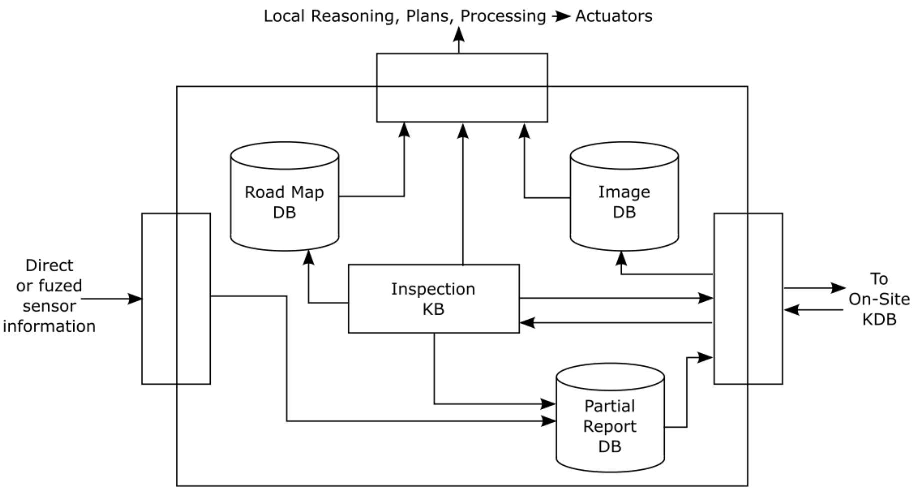

Fig. 5. The preliminary view of the KDB architecture on-board the CFR (D26.21).

We should mention here also, as the client of the KDB, the Inspection Planner module (see Fig 3, D26.12). It follows the high level plans developed for the mission, retrieves the low level (ROS-level) procedures pre-programmed for the high level actions and supervises their execution, passing mobility ROS commands to the Navigation Planner and other ROS commands to the suitable ROS nodes.

3 Knowledge and Database component

3.1 Preliminary discussion

The preliminary concept of the KDB component envisaged the place of the CFR in the whole information system of the industrial facility, see Fig. 4. Under this vision the domain and application knowledge should be provided (downloaded) from the Remote centre to the Plant operator centre, where mixed with some more concrete application knowledge would be used (under plant operator supervision) to develop inspection mission plans and procedures. The plans would be then downloaded to the CFR onboard information system to control the mission execution.

Data (numerical, factual, event-like, messages) collected during the particular mission could be then during the mission or after it uploaded to the plant centre, where it could be analyzed, evaluated, extracted for reference and storage, and in the resume form passed higher in the facility hierarchy.

The preliminary design then called for separate knowledge bases at every level of the hierarchical information system; with separate storage components for various forms of knowledge (Fig. 5, for more detail see D26.21).

The assumed strong connection between the CFR and the Plant operator centre (demanded in the use case at least by the teleoperating option) was also taken into account in the proposal of the CFR Planning and Reasoning Architecture in D26.12 (see also the upper part of the Fig. 3).

Although the Use Case 4.1.3 Co-worker problem is technically well-defined, it entails yet a number of requirements and more specific problems where the algorithmic solutions are not yet set to the final choice. The final choice of the algorithms (especially learning and reasoning in the Learning module) is coupled to the testing results in the real lab environment and is under way. Considering the stringent quality and safety requirements with respect to the acting behavior and the data provided by the CFR, the robot is being tested gradually, firstly under simplified operational conditions. All these issues affect the KBD developments (and this final report) in two aspects.

(1) The CFR testing does not cover yet the live connection with the Plant operator centre software. That information system is outside the present scope of the CFR system design (it is not treated in the use case). There are yet no plans how to embed the CFR related software modules and services into the shift control system.

In consequence all the knowledge required to plan, execute, and evaluate the inspection missions must reside onboard the CFR and be stored in its Knowledge and Database component.

If the policy of the management of the gas-and-oil facility, after the CFR has been tested and operates without errors, will be to make the link between these two system closer in the interest of more streamlined passing of the information, then the original proposal for the IT system hierarchy should be reconsidered.

(2) In the knowledge base of the CFR roughly speaking we have to represent facts, structures, graphs (maps), and images. The question posed in the D26.21 was whether can we force them into a unified mathematical apparatus that permits efficient storage and manipulation and results in a conventional knowledge-based architecture, or it will be better to keep various knowledge chunks represented differently and fuse them together by an integrated architecture?

Knowledge about structures are first of all those of the physical structures of the facility (plant, subunits, equipment), logistic structures (inspection points), and abstract structures (missions, tasks, commands, etc.). Structure knowledge are coupled with plenty of factual information, symbolic and numeric. To store the structured information effectively we proposed to turn to the idea of the original frame-based representation [6, 13, 14].

A special place in the abstract hierarchy of actions belongs to the ROS programming concepts that connect via the ROS middleware the high level concepts of mobility and operability with the actual physical movement and other concrete activities.

Structured information means also semantic maps of the plant, its parts, locations, area shapes, access road graphs, etc. for which the KML based XML descriptors were proposed (see D41.20) [8].

Other available and required information is binary in character. Here we have pixel level maps, images, audio records. The use case (and the preliminary proposals) also accounts for the option of the videostreaming when teleoperating, however this option has not been developed yet to enter the testing and is not taken into account in the final proposal of the KDB.

3.2 Knowledge and Database implementation - final proposal

In the D26.30 a relational database implementation was proposed to realize the Knowledge and Database component. Its rationale is as follows:

- The tradititional knowledge base in the AI sense is a component containing beside the knowledge chunks about the domain (relational information and grounded facts) also reasoning knowledge chunks, i.e. rules, and heuristics shaping the course of the reasoning with rules and facts. From the point of view of the reasoning the co-worker domain is relatively accessible, well structured, changing, but not excessively dynamic. The anticipation of the contingent exogenous events is yet required but the spectrum of the events is limited. This part of knowledge was reviewed on the design table and implemented procedurally as pro-active algorithms to compute the optimal inspection paths and re-active algorithms reacting in a regulated way to a pre-prepared list of unwanted events. In consequence, and also taking into account that the application domain does not change significantly, the knowledge required to run these algorithms is heavily compiled into a common database architecture as the executable model of the application domain (D26.30, D26.40).

- The inspection mission problem domain is of the CWA (Close World Assumption) character which fits well the semantics of a database (or rather it should be made into a CWA to make the problem domain testable).
- The structure and the information elements of the relational database have the describing capability and semantics almost fully consistent with that of the frame based approach. Perhaps the only essential difference is the introduction of the so called daemons (if-needed, if-deleted, etc.…) in the original frame concept. It can be however also realized in a database but with a more walking around and was given up as the current version of CFR being tested now did not required it.
- Database makes it possible to naturally integrate the “knowledge” and the “data” information (i.e. it is a fusion of the Knowledge Base and Working Memory in the original sense of those components). A Working Memory component was needed in the CFR system anyhow.
- Due to the developed interface, a seamless sharing of information by independent system components.
- This implementation choice makes it possible to effectively and flexibly fuse together knowledge in incompatible formats like logic, binary, … It can be done via the own mechanism of the database (e.g. JSON based) or by using separate repositories for the non-logical information and link them to other knowledge by file references.
- Considering that the high level function should seamlessly integrate with the ROS middleware, last but not least is the availability of the ROS nodes implementing rich relational databases.

In the following we review how the knowledge essential to carry on the co-worker inspection missions is mapped to the database architecture. As the co-worker prototype is yet under development, the detailed content of the Knowledge and Database (implemented as relational database) can yet be modified, the general structure of the relations and the set of the main modelled concepts can be considered finalized.

Fig. 6. The structure of the Knowledge and Database component.

3.3 Knowledge elements explained

In the following we review the domain, application and case knowledge intended to be stored in the KDB and present its model as the relational database. We shortly define and discuss the content of the records, providing explanations and occasional examples. When discussing the concepts (records) only their most important content is mentioned. Other link information can be deduced from the Fig. 6.

3.3.1 Domain knowledge

PLANTS: the place where the mission is performed. Used by the planning algorithms (selecting and routing)

- idPlant: unique identifier
- Description: textual description of the plant
- Each plant has its set of inspection point places (the whole list of possible points)
- Each plant has its set of final inspection points (the points selected by the Selector and Router algorithm)
- Each plant has its own plant map
- Each plant has its own emergency path (system)

INSPECTION_POINTS: are the facilities that the CFR must visit to carry out the mission. Used by the planning algorithms (selecting and routing).

- idInspection_points: unique identifier
- x: coordinate x
- y: coordinate y
- Yaw: orientation
- Description: : explanation of the inspection point
- Interesting: in case of time/cost restrictions, this field show the „interest” or „priority” of the inspection point
- Plants_idPlant: each inspection point belongs to a concrete plant
- Each inspection point has its own constraints
- Each inspection point has CFR constraints for the mission
- Each inspection point has its own equipment

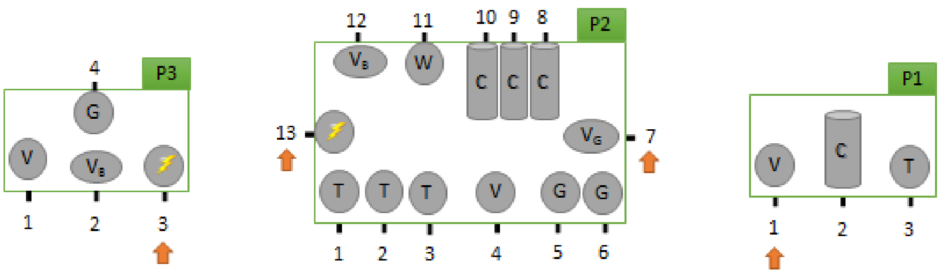

Fig. 7. Inspection points in the lab facility in Trondheim. (D41.20)

INSPECTION_POINTS_CONSTRAINTS: are the limitations or requirements associated with the inspection points that the planning algorithms will fetch and evaluate when they select the correct inspection points to be part of the mission. Used by the planning algorithms (selecting and routing). (see also Section: Open question)

- IdInspectionPointsConstraints: unique identifier
- Description: textual description of the character of the inspection point
- Time_constraint: time limitations for the CFR to observe when inspecting this point (may be violated and summarize to abnormal situation)
- Battery_constraint: battery (spending) limitations for the CFR to observe when inspecting this point (may be violated and summarize to abnormal situation)

CFR_CONSTRAINTS: are the limitations or requirements associated with the CFR when it carries out the mission. Used by the planning algorithms (selecting and routing). (see also Section: Open question)

- IdCFR_constraints: unique identifier
- Capacity1: what the CFR is able to do or not
- Capacity2: what the CFR is able to do or not

Comment: Capacity limits can define applicable or permissible robot dimensions, weight, load capacity, autonomy measured in hours or kilometers, maximum speed, resistance to rain, snowfall, icefloor, vibration, GPS-outage, seeing or not in the dark, resistance or sensitivity to noise, overload, operating range for sensory observations, field of view, spatial and depth resolution of observations, power constraints, size constraints,

EQUIPMENT: the facility specifications on which the CFR will have to work. Used by the planning algorithms (selecting and routing).

- idEquipment: unique identifier of the equipment
- Inspection_Point: parent to the equipment
- Description: textual description of the equipment (e.g. a pressure measuring dial, a water tank, etc.)

EMERGENCY_PATHS: the CFR must follow a route when an alert warns on any problem in the plant. For the CFR purposes.

- idEmergency_paths: unique identifier
- Description: textual description of the emergency path
- coordinates of the path segment to move (i.e. x1, y1, x2, y2, x3, y3, … etc.)

PLANT_MAPS: the structure of he plant as a semantic map. For the CFR purposes.

- idPlant: unique identifier
- idPlant_Map: identifier of the plant map file
- description: explanation of the map

Comment 1: The inspection planner requires explicitly represented and stored data and information for its planning and re-planning activities. Part of it is specific to the planner; however the environment representation has to be shared among the system components as a common reference and grounding basis. This sharable representation will contain:

- The missions described as sequences of actions (e.g. move to waypoint, gather sensor data, ask/wait for operator feedback).
- The inspection points, i.e. the waypoints in the map that represent useful locations from which the robot shall perform the inspection of the relevant equipment.
- The position of different equipment in the environment, linked to some relevant data, such as sensor to be used, expected sensor readings and level of priority.

Elements of the environment required by individual modules and not shared will be stored independently, accessible by the parent module. Such are e.g. the occupancy maps and costmaps used by the Navigation Planner.

Comment 2: For the storage of the inspection points KML representation was chosen (an extended XML notation to express geographic annotations and visualizations in Internet-based, 2D maps and 3D Earth browsers. That way we can use instead of GPS meaningful names for the inspection points and other relevant locations (e.g. charging position). Consider an example (from D41.20):

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Placemark>
  <name>Charging Position</name>
  <description>Outdoor charging position for the Summit XHL robot
  </description>
  <Point><coordinates>74.006393,40.714172,90</coordinates></Point>
</Placemark>
</Document>
</kml>
```

3.3.2 Application Knowledge elements

MISSIONS: is the goal of the CFR. Used by the planning algorithms (selecting and routing).

- idMission: unique identifier
- Description: textual description of the mission
- A mission is composed by several inspection points, e.g.
    - Point1: id_point_1
    - Point2: id_point_2
    - etc.

INSPECTION_POINTS: are the points that form part of the final mission. To be used for the planning and scheduling of the intended mission:

- idFinal_Inspection_points: unique identifier
- x: coordinate x
- y: coordinate y
- Yaw: orientation
- Description: : textual description of the inspection point
- Interesting: in case of time/cost restrictions, this field shows the „interest" or „priority" level of the inspection point
- Plants_idPlant: each inspection point belongs to a concrete plant

Comment: Inspection point has the coordinates relative to the reference frame set in one of the corners of the lab, and distances are measured in meters from there, e.g..

- idFinal_Inspection_points: module_1_gauge_3
- x, y, yaw: [10, 4.5, pi]

THRESHOLDS: value limits not to be exceeded in the measurements made on the inspection point. To be used for the control and re-planning of the actual mission:

- idThresholds: unique identifier
- Inspection_Point: The final inspection point to which the threshold valueas are applicable
- Min: minimum value
- Max: maximum value

FINAL_MISSIONS: is the final goal of the CFR. To be used for the scheduling the execution of the actual mission:

- idFinal_Missions: unique identifier
- Description: textual description of the mission
- Date: date of the mission to be carried out

- A final mission has its own Log to track the tasks and unexpected events
- A final misison has its own Learning data from which the CFR can learn for the future
- A final mission has several final inspection points
- A final mission has its own triggers

Comment: As the picture of what a mission may be in general is necessarily less restricted than the actual mission respecting various constraints and perhaps re-planned due to the unanticipated events, the records for mission and inspection points are conceptually doubled. „Mission" and „Inspection Points" mean application level knowledge, „Final Mission" and „Final Inspection Points" mean case level knowledge, a task to be actually carried on. The very same descriptor of Mission and Inspection Points can lead, under different circumstances, to different Final Mission and Final Inspection Points. In consequence the descriptions of the Final Mission and Final Inspection Points are richer.

FINAL_INSPECTION_POINTS: are the definitive points that form part of the final mission. To be used for the scheduling the execution of the actual (i.e. final) mission:

- idFinal_Inspection_points: unique identifier
- Final_Mission: identifier of the final mission addressing this inspection point
- x: coordinate x
- y: coordinate y
- Yaw: orientation
- Description: : textual description of the inspection point
- Interesting: in case of time/cost restrictions, this field shows the „interest" or „priority" level of the inspection point
- Plants_idPlant: each inspection point belongs to a concrete plant
- Each final inspection point has its own tasks
- Each final inspection point has its own thresholds

TRIGGERS: is the event when something abnormal occurs

For the CFR purposes

- idTrigger: unique identifier of the trigger
- Description: the circumstances of the trigger execution and explanation (i.e. time stamp, inspection point/ equipment or the road segment where and when the trigger happened, the type of the trigger, additional parameters and measurements)

TASKS: the inspection task that should be carried out by the CFR at the inspection point.

- idTasks: unique identifier
- Description: textual description of the task
- Each task has a type
- Each task has several commands

TYPE_TASKS: type of a task

- idType_tasks: unique identifier
- Description: textual description of the type of a task (e.g. reading the value of a measuring gauge, check gas leaks)

COMMAND: specifies the high level actions to carry out the mission.

- idCommand: unique identifier
- Description: textual description of the command
- Each Command refers to a ROS_Commands

Comment: Commands are actions dispatched to the robot system that the robot shall execute. Some examples of such actions are "*navigate_to (waypoint X)*" or *"capture_data (IR_cam)"*. High level commands are not executable directly. The Inspection Planner upon receiving a high level command belonging to the mission must fetch the ROS commands and messages implementing the high level command and pass them to the suitable ROS nodes.

ROS_COMMANDS: is the low level command ready to pass for the execution to the middleware.

- idROS_COMMANDS: unique identifier of the ROS_command
- Description: textual description of the ROS_command
- ROS_Command: the ROS_command
- ROS_Command_type
- ROS_Message_type_sent
- ROS_Message_type_received
- ROS_Message_sent

Comment: ROS commands can be, among others, wapoint navigation messages, measurement hardware activating messages, text-to-speech generating messages, alert sound messages, light indicator messages (see D41.20 for detail).

ROS_COMMAND_TYPE: type of different commands

- idROS_command_type: unique identifier
- Description: textual description of the type

ROS_MESSAGE: is the low level message.

- idROS_message: unique identifier
- Description: textual description of the ROS_message
- Message values

Comment: ROS messages can be, among others, odometry messages, battery level indications, messages summarizing the robot and the mobile base diagnostics, robot status and connectivity status messages, messages activating/deactivating emergency navigation mode, plant status messages, “mission to run” message from the GUI, messages tracking obstacles (pedestrians or cyclists) (for more detail see D41.20).

ROS_MESSAGE_TYPE: is the type of low level message.

- idROS_message_type: unique identifier
- Description: textual description of the ROS_message (e.g. topic, service request, service reply, actionlib feedback, actionlib result)

3.3.3 Case Knowledge elements

LOGS: the place where the CFR writes all the issues that happened during the mission, essentially the working memory of the CFR.

- idLogs: unique identifier
- Description: textual description of the log
- Images: subrecord structure to store pictures taken by the CFR, with time stamps and location reference (inspection point/emergency road segment or GPS coordinates)
- Events/Triggers: in principle triggers that happen during the mission, with time stamps and location reference (some other non-trigger events can also be logged, but the information essential to the operator are basically covered by the set of the triggers)

LEARNING: mission data that will be used by the CFR as training examples in learning for future occasions

- idLearning: unique identifier
- Event-descriptor: event location (inspection point/emergency road segment or GPS coordinates), time stamp,
  event coding:
    battery_event: yes/no,
    weather_event: yes/no,
    abnormal_measurement: yes/no,
    trigger_1_launched: yes/no,
    trigger_2_launched: yes/no,
    …

4 Conclusions and open questions

In the present configuration the integration and the communication of the CFR with the Plant operator centre is not tested, beside testing the visualization of the CFR motions required to evaluate the teleoperation option. A number of issues related to the integration of the both system remains open and must be solved only in the future.

4.1 The choice of database

In the present version the CFR KDB is based on PostgreSQL database, as it offers a good balance between functionality and system requirements. The CFR operating characteristics show however some peculiarities which may in the future indicate the need to reconsider the database design underlying the robot KDB.

The amount of data to be stored is not large. There are practically no multiple independent clients, querying in parallel massive amount of real-time data, competing for access, and requiring transactional mechanism to keep consistency. The majority of data has to be discarded on the daily basis. Large data objects are almost solely binary maps and images (videostreams?), as well as XML based files, best to be kept in their native formats. JSON objects resembles greatly the construction style of the ROS messages.

In consequence No SQL, JSON based database design could be competitive, regarding that JSON format ensures that changes in message structure, which are inevitable during the development of a robotics system, will not be breaking changes for the datastore. JSON is structured in a very similar manner (nested dictionaries) to that of ROS messages allowing for simple conversion between the formats. JSON can be queried against as opposed to traditional binary message representations.

4.2 The final prototype

The final prototype of the co-worker robot, being designed by teams at SINTEF and TECNALIA is close to be ready for testing. There are however some issues with the implementation of the planning algorithms, which aren't yet definitely resolved (about the algorithms in question see D26.30 and D26.40) and are related to some concepts stored in the Knowledge abd Database, mainly the „Inspection point constraints" and the „CFR constraints" frames (tables).

The actual knowledge content to be coded is in itself clarified. The topics being yet under discussion is the particular form of the constraint expressions which will serve the best the requirements of the algorithms.

Inspecting a given inspection point draws from the battery capacity and from the inspection time devoted for the mission. As even the same kind of the inspection point placed at different places may mean different access restrictions, the battery and time costs of visiting an inspection point are important parameters to the optimal path planning algorithms.

The type of the inspection point is related also to the applicable measuring equipment (sensors). If such sensor is malfunctioning then the inspection point cannot be fully inspected and this information may influence the planned or re-planned course of the inspection route. To the pecularities of this application domain belongs also the fact that there may be a dependency among the inspection points, e.g. an another inspection point may be specified to be inspected before the current one, or another inspection point must be inspected after the current one, if the current measurements may indicate a malfunction.

As mentioned above these knowledge chunks will be placed in the „Inspection point constraints" or „CFR constraints" slot of the Knowledge and Database, in the final expression format developed for the running prototype.

4.3 Scaling up the KDB with useful services or as yet not implemented features

It is always a critical issue how well the present design of an intelligent system can stand the demands for extensions, improvements, modifications, being "evidently easy" to the final user, but not at all so to the system designer. The more successful the design and the test results, the higher the chance that the final user will come with the new "minute" requirements.

From that point of view (and taken into account that whatever the new demands of the final user) the format of the domain knowledge and at least a part of the application knowledge introduced in the KDB seems hopefully general and flexible enough to accept any future extensions and changes.

4.4 Storing and retrieving the experience of learning

The CFR will learn from the bad experience, i.e. from the situations where unanticipated events happened, originated in the unpredictable environment, malfunctioning of the external facilities, or in the diagnosed problems in the CFR hardware.

All such information will be logged, but the log format, designed for an easy human evaluation, is not necessarily machine learning friendly where labelled training examples are rather needed. For that reason the proposed KDB design makes the redundant choice of storing the critical event data in two record formats, one as the log record, and another as the learning training example record. That way the learning can skip the laborious phase of identifying, lifting and preparing the training examples from the data.

4.5 Post-processing of the KDB

As mentioned earlier the mature CFR system must be earlier or later integrated into the whole facility information system, i.e. it must be interfaced with communication channels to the Plant operation centre, with options of being controlled remotely (see also D26.21). This situation was assumed based on the general trends reported in the literature and the original use case of the co-worker, and led to a preliminary proposal of an architecture with a number of CFR related modules (GUI, storage, control, communication, teleoperation) fused into the Plant operation centre software.

In the present state of testing both system components (i.e. the CFR and the Plant centre) are as yet disconnected and every data processing, during or between inspection missions is run on the CFR.

In the long-run however certain knowledge intensive but not real-time-critical tasks, like machine learning, trend evaluation, computing statistics, computing advanced mission plans, etc. could be relocated to the Plant centre, freeing the CFR software from the unnecessary load and the clash with the regular inspection duties. Due to the sophisticated character of such tasks, their results should be anyhow cross-checked by the human operators.

The question remains then how the large volume of the data should be transferred to the Plant centre for processing. The most feasible solution seems to maintain at the both sides of the communication channel Knowledge and Database components of the same structure (i.e. duplicating the CFR KDB at the Plant centre), to compress it after the inspection mission is concluded and to upload it to the Plant centre, where suitable algorithms can be designed to do the processing off-line.

5 References

[1] **Affordance**, http://en.wikipedia.org/wiki/Affordance

[2] Awaad I., G.K. Kraetzschmar, J. Hertzberg, **Affordance-based reasoning in robot task planning**, Planning and Robotics (PlanRob) Workshop ICAPS-2013.

[3] Barz T., Bonow G., Hegenberg J., Habib K., Cramar L., Welle J., Schulz D., Kroll A., Schmidt L., **Unmanned Inspection of Large Industrial Environments - Insights into Research Project RoboGasInspector**, in N. Aschenbruck, P. Martini, M. Meier, J. Tölle, ed., Future Security, Vol. 318, Comm. in Computer and Information Science, Springer, Bonn, pp. 216-219, Sept 4-6, 2012.

[4] Correll, N. and Martinoli, A., **Multirobot inspection of industrial machinery**, Robotics & Automation Mag., IEEE , vol.16, no.1, pp.103-112, March 2009

[5] Distante C., G. Indiveri and G. Reina, **An application of mobile robotics for olfactory monitoring of hazardous industrial sites**, Industrial Robot: An Int. Journal, 36/1 (2009) 51–59

[6] Fikes R. and T. Kehler, **The Role of Frame-based Representation in Reasoning**, Comm. of the ACM, Sept 1985, Vol 28, No 9, 904-920.

[7] Habib A., G. Bonov, A. Kroll, J. Hegenberg, L. Schmidt, T. Barz, D. Schulz, **ROBOGAS$^{\mathbf{INSPECTOR}}$ Research Project: Detecting Gas Leaks with Autonomous Mobile Robots**, Ex-MAGAZINE 2014, pp. 91-97

[8] **Keyhole Markup Language**, https://en.wikipedia.org/wiki/Keyhole_Markup_Language

[9] Kroll, A., **A survey on mobile robots for industrial inspection**, Int. Conf. on Intelligent Autonomous Systems (IAS10), pp. 406-414, 23 - 25 July, 2008.

[10] Kumar V., D. Rus, S. Singh, **Robot and Sensor Networks for First Responders**, PERVASIVEcomputing, Oct–Dec 2004, pp. 24-33

[11] Maurtua I., L. Susperregi, A. Fernández, C. Tubío, C. Perez, J. Rodríguez, T., Felsch, M. Ghrissi, **MAINBOT – mobile robots for inspection and maintenance in extensive industrial plants**, Energy Procedia 49 (2014) 1810 – 1819, SolarPACES 2013

[12] Mulder J., X. Wang, F. Ferwerda, M. Cao, **Mobile Sensor Networks for Inspection Tasks in Harsh Industrial Environments**, Sensors 2010, 10, 1599-1618

[13] Petersen W., **Representation of Concept as Frames**, The Baltic Int. Yearbook of Cognition, Logic and Communication – Vol. 2, 151-170, Univ. of Latvia, Riga, 2007.

[14] Rattanaprateep, C. ; Chittayasothorn, S., **A frame-based object relational expert database system**, AFRICON 2007, 26-28 Sept. 2007, Windhoek, pp. 1-7.

[15] Sahin, E., Cakmak, M., Dogar, M.R., Ugur, E., Ucoluk, G.: **To Afford or Not to Afford: A New Formalization of Affordances Toward Affordance-based Robot Control**. Adaptive Behavior 15(4), 447–472, 2007.

[16] **Sensabot: A Safe and Cost-Effective Inspection Solution,** J. of Petroleum Techn., www.jptonline.org, No 10, Vol 64, Oct 2012OBER 2012

[17] Soldan, S.; Bonow, G. and Kroll, A., **RoboGasInspector – A Mobile Robotic System for Remote Leak Sensing and Localization in Large Industrial Environments: Overview and First Results**, Proc. of the 2012 IFAC Workshop on Automatic Control in Offshore Oil and Gas Production, pp. 33-38, 31.05 - 01.06., 2012.