



Page 1 of 12
Date: 31.10.2016
Dissemination Level: PU
610917 – STAMINA



Sustainable and reliable robotics for part handling in manufacturing

Project no.: 610917

Project full title: Sustainable and reliable robotics for part handling in manufacturing

Project Acronym: STAMINA

Deliverable no.: D4.4.1

Title of the document: Report on the Specification of a Mission Planning Domain for Robot Fleet Behaviour Control

Contractual Date of Delivery to the CEC:	31.10.2016
Actual Date of Delivery to the CEC:	31.10.2016
Organisation name of lead contractor for this document:	Heriot-Watt University (HWU)
Author(s):	Matthew Crosby, Ron Petrick
Work package contributing to the document:	WP4
Nature:	R
Version	1.0
Total number of pages:	12
Start date of project:	01.10.2013
Duration	42 months – 31.03.2017

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 610917

Dissemination Level

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

This deliverable reports on the planning domain used by the mission planner for producing mission assignments to control individual robots in the robot fleet. The planning domain uses information about kitting orders, robot skills, and the state of the operating environment provided by the logistic planner. Generated mission assignments are returned to the logistic planner which ultimately relays them to the individual robots. This deliverable presents a status report on the mission planner and the planning domain description used in STAMINA.

Keyword List: Mission planner, planning domain, automated multiagent planning, high-level architecture

Document History

Version	Date	Author (Unit)	Description
1.0	20.10.2016	M.Crosby (HWU), R.Petrick (HWU)	Initial version of document.

Contents

1	Executive summary	4
1.1	Overview and background	4
1.2	Main contributions and developments	5
1.3	Ongoing and future work	5
2	High-level STAMINA system architecture	6
3	Mission planning architecture and services	7
3.1	MissionPlanner_PlanKittingOrders	7
3.2	MissionPlanner_ParkRobots	7
3.3	LogisticPlanner_DispatchMissions	8
4	Mission planning domain and problem descriptions	8
4.1	PDDL domain description	8
4.2	PDDL problem description	8
5	Mission and task planning in the STAMINA system	11
	References	12

1 Executive summary

1.1 Overview and background

A central objective of WP4 (Mission tasks and vertical enterprise integration) is to support the integration of a collection of robots with different skills into a heterogeneous robot fleet, by defining and implementing the architecture for a *mission planning subsystem*. The mission planner itself is responsible for generating mission assignments which map particular kitting orders to individual robots in the fleet. The mission planning architecture must therefore provide the necessary interfaces for communicating with the logistic planner, which supplies information about the status of the kitting orders, robots, and the environment, and for returning completed mission plans to the logistic planner for delivery to the robot fleet.

High-level mission planning capabilities in STAMINA are supplied by the ADP planner (Agent Decomposition Planner), which UEDIN is developing for use with multiple robots in a real-world factory environment. ADP is a multiagent automated planner which is designed to deduce assignments of goals and actions to particular agents (robots) from standard Planning Domain Definition Language (PDDL) encodings of the planning problem, and to exploit these decompositions for efficient planning. The PDDL encoding makes use of a *domain description*, which models the dynamics of its operating environment, and a *problem description*, which encodes the specific goal the planner is attempting to achieve.

Like most automated planners, ADP operates best in discrete, symbolic state spaces described using logic-like representations such as PDDL, the de facto standard representation of the automated planning community. As a result, work that addresses the problem of integrating automated planning on real-world robot systems often centres around the problem of representation, and how to abstract the capabilities of the robot(s) and their working environment so that they can be put into a suitable form for use by a goal-directed planner. Integration also requires the ability to communicate information between system components. Thus, the design of the mission planning subsystem must take into consideration external concerns, to ensure proper interoperability with modules that aren't traditionally considered part of the core planning process.

In the first year of STAMINA (Month 1–Month 12), the work of WP4 considered the design of the high-level architecture of which the mission planning subsystem was a main component. An initial version of this architecture was presented in deliverable D4.1.1 (Specification of SOA for robot fleets). In the second year of project (Month 13–Month 24), this interface was further refined and an updated version of the high-level STAMINA architecture was presented in deliverable D4.2.3 (Conversion logic and high level operations scheduling). Likewise, an initial prototype of the mission planner was developed for offline testing during Year 1, and was integrated as part of the STAMINA system for initial online testing during Year 2. Deliverable D4.3.1 (Architecture Description for the Mission Planning Subsystem) reported on the state of the mission planning architecture, and provided details of the implementation of the mission planner itself, at Month 27 of the project. Since that time, there have been no major changes to the mission planner itself, except for minor performance enhancements and software bugfixes.

This deliverable complements D4.3.1 by focusing on the PDDL definition of the planning domain and problem used by the mission planner to produce mission assignments at Month 36 of the project.

1.2 Main contributions and developments

The main contribution of this deliverable is a description of the PDDL domain and problem definitions used by the mission planner to generate mission assignments on STAMINA. Overall, this deliverable reports on a number of significant developments:

- A brief description of the current high-level architecture featuring the logistic planner, mission planner, skills framework, and task planner. While in previous versions of the STAMINA system the mission planner not only produced a mission assignment for individual robots in the fleet, but also a plan for how each of those robots could achieve its goals. In the current version of the system, the mission planner only produces mission assignments with the goal of fulfilling those assignments passed to the individual robots' task planner, thereby achieving the original planned functionality for the mission planner. The associated Month 36 deliverable D3.4 (Skill-Based Task Planning) reports on the task planning process in greater detail.
- The mission planner was successfully tested during the third test sprint. The mission planner component successfully interacted directly with the logistic planner (and indirectly with the skills framework) and returned appropriate mission assignments for all problems tested during the test sprint.
- Testing has continued with the ADP multiagent planning algorithm in simulation to solve larger instances of STAMINA-like problems that go beyond the physical limitations of the single robot system being tested in the real factory environment. It is expected that the multiagent testing will continue in the final phase of the problem, with integration planned with a simulation environment.

1.3 Ongoing and future work

A number of tasks remain open at the time of this report and constitute ongoing and future work:

- The current version of the mission planner now separates the task of mission assignment from the task of plan generation to achieve those missions, with the latter task being performed by the task planner (WP3). However, large scale testing (in simulation) is still needed to properly evaluate this separation when scaling to larger numbers of robots and larger factory sizes.
- Further collaboration between Task 4.4 and Task 2.5 for multi-robot coordination is now possible given the separation of mission and task planning described above. Simulated testing will now be able to include all components of the STAMINA system. In particular, the interaction between different design choices in the task planner and mission planner components can be tested.
- The multiagent planning algorithms used by the mission planner will continue to be tested and improved in the remainder of the project.

In the remainder of this deliverable we highlight the current high-level architecture used on STAMINA, describe the main services provided by the mission planner, outline the planning domain definition used to generate mission assignments in the system, and briefly discuss the connection between mission planning and task planning.

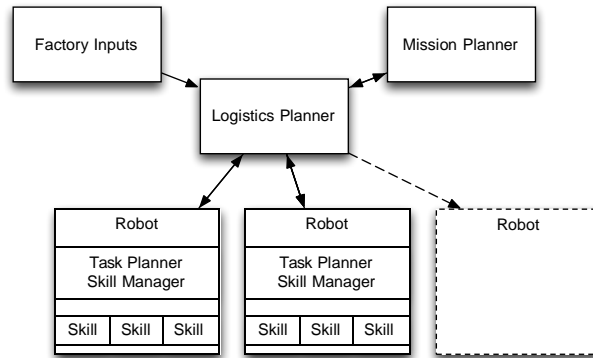


Figure 1: High-level STAMINA system architecture

2 High-level STAMINA system architecture

Figure 1 shows the high-level system architecture in STAMINA with the central logistic planner, mission planner, and individual skill-based robot with their task planners. The logistic planner and mission planner are being developed as part of WP4, while the skills framework and task planner are part of WP3. We briefly summarise the operation of these various components below.

Logistic Planner: The logistic planner is responsible for integrating the STAMINA system with the factory’s information systems, providing the necessary conversion logic between data provided by the factory system and STAMINA components. Data provided by the logistic planner is supplied in the form of a *world model* which includes information about the operating environment, including the identification of physical objects (racks, small and large boxes, conveyors, kits, and parts) and their 3D models. The logistic planner also provides information about the status of the robots in the fleet and their capabilities. Finally, the logistic planner is responsible for managing *kitting orders* originating in the factory systems: lists of parts that must be collected in boxes called *kits* for delivery to the assembly line.

Mission Planner: The mission planner is responsible for generating mission assignments for individual robots in the fleet. Missions are created based on the robot capabilities and the set of available kitting orders, and are defined as a goal assignment to a specific robot. In our simulated environment, each robot can carry two kitting boxes so a mission assignment may involve the completion of up to two kitting orders per robot. The mission planner interacts with the logistic planner to obtain the world model that provides details of the current set of kitting orders to be completed, the state of the robots in the fleet and their available skills, and information about the robots’ operating environment (e.g., the map and location of objects in the kitting area). This information is used to build a PDDL planning domain and problem description. Missions are then constructed using the multiagent decomposition planner ADBP [Crosby et al., 2013, Crosby and Petrick, 2015] which generates a sequence of skills for each robot to execute to fulfil its kitting order(s). Mission assignments are returned to the logistic planner which is responsible for delivering them to the individual robots in the fleet. More information about the mission planner can be found in deliverable D4.3.1 (Architecture Description for the Mission Planning Subsystem).

Skills Framework: The *SkiROS (Skills-ROS)* [Rovida and Krüger, 2015] skills framework is a modular and scalable robot software architecture that extends the Robot Operating System (ROS). Robot skills can be seen as the high-level building blocks from which a complete robot task can be composed. Skills are object-centred robot abilities that, unlike traditional robot programs, include references to physical objects instead of, e.g., 3D coordinates. Using sensory devices, the robot employs object detection functions to identify the concrete “low-level” parameters that correspond to the high-level objects. Similar to planning actions, SkiROS skills are modelled with preconditions that must be established before a skill can be executed, and postconditions that are achieved by executing the skill.

Task Planner: Mission assignments from the mission planner are passed to individual robots through the logistic planner, for execution by the skills framework. All robot-specific planning (e.g., initial plan generation and replanning in the event of plan failure) is performed on the robot by the task planner, to find executable skill sequences. To do so, the task planner creates a planning domain and problem in PDDL directly from the robot’s skills definition and world model information, using goals specified in the mission assignment. Plans are built using the FastDownward planner [Helmert, 2006] with A* search and the landmark-cut heuristic. More information about the task planner can be found in deliverable D3.4 (Skill-Based Task Planning).

3 Mission planning architecture and services

At a high level, the mission planner’s role is to compute task assignments (goals) for the robot fleet. Assuming maximum functionality, such assignments might require reasoning about multiple robots, taking into consideration the collaboration requirements for individual kits, in order to utilise each robot’s skills most efficiently. However, for the purpose of testing on the project, there is only one robot, therefore the mission planner can be run in a reduced mode which is sufficient to support the constraints of the actual deployment environment.

In the STAMINA architecture, the mission planner only communicates with the logistic planner. This allows for standardisation of the communication system so that the robot fleet does not require a separate communication protocol for interacting with the mission planner and the logistic planner. This also allows the logistic planner to be the sole access point for workers and technicians, streamlining their control over the system.

3.1 MissionPlanner PlanKittingOrders

This service is provided by the mission planner and it takes all current kitting orders available to the logistic planner along with the logistic planner world model. It returns either ‘accepted’ or ‘refused’ depending on if it detects any inconsistencies in the kitting order list or the world model. It is defined as:

```
KittingOrder[] kittingOrders
Node[] nodeList
---
ActionAcceptedRefused actionAcceptedRefused
```

Once accepted, the mission planner recreates the world model in PDDL format and attempts to find plans for the set of kitting orders. The output plans are converted into missions and returned to the logistic planner using the DispatchMissions service described below.

3.2 MissionPlanner ParkRobots

This service was added more recently to the mission planner as a way for the logistic planner to request that a set of robots are returned to their docking stations. This will be called at the end of the day, or when the factory floor needs to be cleared for some reason. The ParkRobots service is defined as follows:

```
string[] robot_ids
Node[] nodeList
---
ActionAcceptedRefused actionAcceptedRefused
```

As with all services, it is returned whether or not the service is accepted, and if not, an error message is generated and returned. For example, the error message might be that no robot was found with the specified id in the world model. This service triggers the mission planner to run its planner to make sure that all robots can be returned to their docking locations and then the appropriate missions are returned to the logistic planner.

3.3 LogisticPlannerDispatchMissions

This service is provided by the logistic planner and allows the mission planner to send missions that it has created. It is defined as:

```
Mission[] missions
---
ActionAcceptedRefused actionAcceptedRefused
```

where Mission[] is an ordered list of all missions to be achieved.

Missions are defined as:

```
string[] kittingOrderIdList
string   robotId
string[] goals
```

where a single robot is given a set of goals which will be sent to the SkiROS component of the STAMINA system. It is also reported which kitting orders the mission intends to fulfil, which is used by the logistic planner to keep track of the system.

4 Mission planning domain and problem descriptions

The Planning Domain Definition Language (PDDL) [McDermott et al., 1998] is the de facto standard planning representation language, used by most planning algorithms and the International Planning Competitions. By converting the STAMINA world model into PDDL format, the planning algorithm used by the mission planner can be easily updated as planning technology advances.

PDDL files come in two parts. The first part is the domain file which defines the type of actions in the planning scenario. The second part is the problem file which defines the objects in the domain and the goals that need to be achieved. The domain file remains the same for each call to the mission planner while the problem file is automatically generated from the world model and kitting orders provided by the logistic planner.

4.1 PDDL domain description

Because the mission planner is designed to handle multiple robots it is important to have an efficiently designed planning domain. Therefore, the planning domain for the mission planner was designed by hand and uses some compilation techniques to reduce complexity and allow for better scaling as the number of robots and size of the domain increases. For example, while the robots are defined as having separate pick and place actions, these are combined into a single pick-place action which has their combined effect. This does not change which goals are reachable because in the planning domain pick is always followed by a place command as kitting orders always require parts to be placed in a certain location. The domain file is shown in Figure 2.

4.2 PDDL problem description

The problem file is automatically generated from the world model and changes each time. A small example problem file with two robots and two achievable kitting orders of size three is shown in Figure 3.

Note that the STAMINA names of objects are changed to be consistent with standard planning conventions. These are converted back before being sent back to the logistic planner.

```

(define (domain stamina)
  (:requirements :typing)

  (:types
    cell kit robot part location
    box - location
    smallbox largebox - box
  )

  (:predicates
    (robot-at ?r - robot ?l - location)
    (carrying ?r - robot ?k - kit)
    (in-kit ?p - part ?k - kit ?c - cell)
    (cell-component ?c - cell ?k - kit)
    (fits ?p - part ?c - cell)
    (in-box ?p - part ?b - box)
  )

  (:functions
    (total-cost)
  )

  (:action drive
    :parameters (?r - robot ?x - location ?y - location)
    :precondition (and
      (robot-at ?r ?x)
    )
    :effect (and
      (not (robot-at ?r ?x))
      (robot-at ?r ?y)
      (increase (total-cost) 10)
    )
  )

  (:action pick-place
    :parameters (?r - robot ?p - part ?b - box ?c - cell ?k - kit)
    :precondition (and
      (robot-at ?r ?b)
      (in-box ?p ?b)
      (cell-component ?c ?k)
      (carrying ?r ?k)
      (fits ?p ?c)
    )
    :effect (and
      (in-kit ?p ?k ?c)
      (increase (total-cost) 1)
    )
  )
)

```

Figure 2: Mission planner domain file in PDDL

```

(define (problem staminal) (:domain stamina)
  (:objects
    robot0 robot1 - robot
    loc0 loc1 - location
    largebox1 - largebox
    smallbox1 smallbox2 smallbox3 smallbox4 smallbox5 smallbox6 - smallbox
    part0 part1 part2 part3 part4 part5 part6 - part
    kit0 kit1 - kit
    kit0c0 kit0c1 kit0c2 kit1c0 kit1c1 kit1c2 - cell
  )

  (:init
    (robot-at robot0 loc0)
    (carrying robot0 kit0)
    (robot-at robot1 loc1)
    (carrying robot1 kit1)
    (in-box part0 smallbox1)
    (in-box part1 smallbox2)
    (in-box part2 smallbox3)
    (in-box part3 smallbox4)
    (in-box part4 smallbox5)
    (in-box part5 smallbox6)
    (in-box part6 largebox1)
    (cell-component kit0c0 kit0)
    (cell-component kit0c1 kit0)
    (cell-component kit0c2 kit0)
    (cell-component kit1c0 kit0)
    (cell-component kit1c1 kit0)
    (cell-component kit1c2 kit0)
    (fits part0 kit0c0)
    (fits part2 kit0c1)
    (fits part4 kit0c2)
    (fits part4 kit1c0)
    (fits part1 kit1c1)
    (fits part2 kit1c2)
    (= (total-cost) 0)
  )

  (:goal (and
    (ObjectInCell kit0c0 part0)
    (ObjectInCell kit0c1 part2)
    (ObjectInCell kit0c2 part4)
    (ObjectInCell kit1c0 part4)
    (ObjectInCell kit1c1 part1)
    (ObjectInCell kit1c2 part2)
  ))
)

```

Figure 3: A mission planner problem described in PDDL

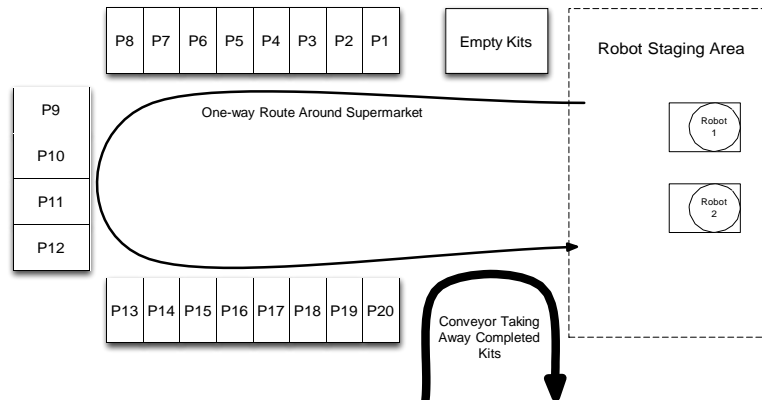


Figure 4: An abstract view of the factory warehouse environment.

5 Mission and task planning in the STAMINA system

Planning is performed at two levels in the STAMINA system: at a high level by the mission planner to find mission assignments for the robot fleet; and by each robot using its on-board task planner and automatically generated domain model, to find sequences of skills that achieve its goals. The domain setting of the factory warehouse environment is similar to that of Figure 4, with multiple robots and parts. Robots must follow fixed paths through the factory and each robot has space to carry up to two kits, where a kit is a box with a number of compartments, each designed to fit a specific part. Using its PDDL domain and problem definition, the mission planner provides goals for each of the robots in the fleet so that the kits can be completed within a specific time frame. The robots' onboard task planners (WP3) then generate individual plans as sequences of skills to achieve their respective goals (missions). Available skills include actions like NAVIGATE, PICK, and PLACE.

For instance, a plan generated by the task planner for a robot to fulfil two kitting orders assigned by the mission planner may have the form:

```
001: NAVIGATE(robot1, kit2, loc4)
002: PICK(robot1, part1, box4)
003: PLACE(robot1, part1, kit1)
004: NAVIGATE(robot1, loc4, loc12)
005: PICK(robot1, part5, loc12)
016: PLACE(robot1, part5, kit2)
...
031: NAVIGATE(robot1, loc45, locoutput)
```

The mission planner has been tested online in the factory environment during the test sprints, with additional testing currently underway. However, due to the limitations of real-world testing, this domain was restricted to a single robot with only a few parts available. For future demonstrations, we will also use a simulated environment with multiple robots and more parts to illustrate the behaviour of the high-level components of the system, as described above.

References

- [Crosby and Petrick, 2015] Crosby, M. and Petrick, R. (2015). Mission planning for a robot factory fleet. In *IROS 2015 Workshop on Task Planning for Intelligent Robots in Service and Manufacturing*, Hamburg, Germany.
- [Crosby et al., 2013] Crosby, M., Rovatsos, M., and Petrick, R. (2013). Automated agent decomposition for classical planning. In *Proc. of the International Conference on Automated Planning and Scheduling (ICAPS)*.
- [Helmert, 2006] Helmert, M. (2006). The Fast Downward Planning System. *JAIR*, 26:191–246.
- [McDermott et al., 1998] McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). PDDL – The Planning Domain Definition Language (Version 1.2). Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control.
- [Rovida and Krüger, 2015] Rovida, F. and Krüger, V. (2015). Design and development of a software architecture for autonomous mobile manipulators in industrial environments. In *Int. Conf. on Industrial Technology (ICIT)*.