

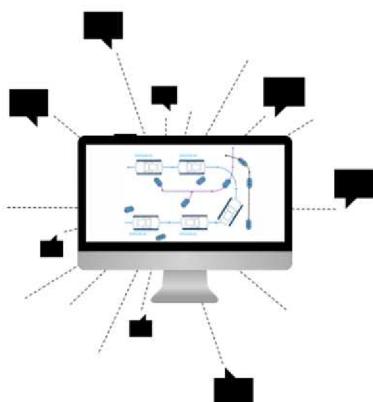


VDA 建议

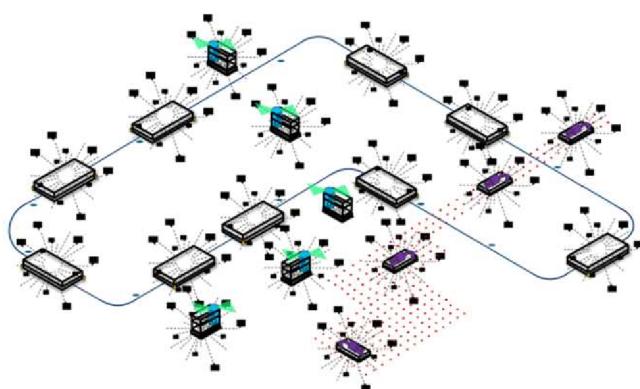
自动导引车 (AGV) 与主控之间的通信接口

VDA 5050

Version 2.1.0, January 2025



control system



automated guided vehicles

无人驾驶运输系统（DTS）通信接口的定义。本建议描述了用于内部物流流程的中央主控和自动引导车辆（AGV）之间交换订单和状态数据的通信接口。

免责声明

以下说明作为在自动导引车（AGV）与主控之间执行通信接口的参考，适用于所有人且无约束力。采用者应确保在具体情况下正确应用。

他们应当考虑当时适用的技术状态。通过应用这些提议，没有人可以逃避自己行为的责任。这些声明不声称是详尽的或对现有立法的精确解释。它们不能替代对相关政策、法律和法规的研究。此外，还应当考虑各自产品的特殊特征以及它们不同的可能应用。在这方面，每个人都应自行承担风险。VDA 及其参与提议开发或应用的人员的责任被排除。

如果你在应用这些提议或可能存在错误解释时遇到任何不准确之处，请立即通知 VDA，以便任何缺陷都能得到纠正。

Publisher Verband der Automobilindustrie e.V. (VDA) Behrenstraße 35, 10117 Berlin,
Germany www.vda.de

Copyright Association of the Automotive Industry (VDA) Reproduction and any other form of reproduction is only permitted with specification of the source.

免责申明

VDA 建议是任何人都可以自由采用的建议。用户负责根据具体情况正确实施建议。

建议考虑了出版时现有的技术。使用 VDA 建议不会免除任何人的责任，所有用户自行承担风险。VDA 和参与起草 VDA 建议的人员的责任被排除在外。

目录

1 前言	5
2 文档目的	5
3 范围	6
3.1 其他适用文档	7
4 要求和协议定义	7
5 通信的内容和过程	8
6 协议规范	10
6.1 表格符号及格式含义	10
6.2 MQTT 连接处理、安全性和 QoS	11
6.3 MQTT 主题层级	11
6.4 协议头	12
6.5 通信主题	13
6.6 主题: "order" (从主控到 AGV)	14
6.7 Maps	29
6.8 动作	33
6.9 主题: "instantActions" (从主控到控制到 AGV)	42
6.10 主题: "state" (从 AGV 到主控)	42
6.11 动作状态	54
6.12 动作阻塞类型和顺序	55
6.13 主题 "visualization"	57
6.14 主题 "connection"	57
6.15 主题 "factsheet"	58
7 Best practice 最佳实践	68
7.1 错误引用	68
7.2 参数格式	68

8 术语表 69

8.1 定义	69
--------	----

图列表

图 1 DTS 库存系统集成	6
图 2 信息流结构	8
图 3 主控中的图表示和订单中传输的图	14
图 4 变更行驶路线 "Horizon" 的流程	15
图 5 订单的伪代码	16
图 6 订单更新的伪代码。注意 <code>orderUpdateId</code> 的变化	16
图 7 定期更新流程 - 订单扩展	17
图 8 接收订单或订单更新的过程	18
图 9 取消订单后的预期行为	20
图 10 具有 <code>corridor</code> 属性的边缘，定义了允许车辆偏离其预定义轨迹以避开障碍物的左右边界。左侧，运动中心定义了允许的偏差，而右侧，车辆的轮廓（可能因负载而延伸）定义了允许的偏差。这是由 <code>corridorRefPoint</code> 参数定义的。	22
图 11 坐标系示例及 AGV 方向	29
图 12 地图和车辆的坐标系	30
图 13 主控、AGV 和地图服务器之间下载、启用和删除地图所需的通信	31
图 14 状态主题提供的订单信息。仅传输最后一个节点的 ID 以及剩余的节点和边	43
图 15 订单处理期间 <code>nodeStates</code> 、 <code>edgeStates</code> 、 <code>actionStates</code> 的示意图	44
图 16 动作状态所有可能的转换	55
图 17 处理多个动作	56

表列表

表 1 操作模式及其含义	54
表 2 <code>actionStatus</code> 字段的接受值	54
表 3 动作阻塞类型	55

1 前言

该接口是在德国汽车工业协会 (VDA) 和 德国机械设备制造业联合会 (VDMA) 合作下建立的。双方的目标是创建一个通用的接口。对接口的修改建议应提交给 VDA，与 VDMA 共同评估，并在通过决策后纳入新版本状态。通过 GitHub 对本文档的贡献深表感谢。仓库可在以下链接找到：<https://github.com/vda5050/vda5050>.

2 文档目的

推荐的目标是简化新车辆与现有主控制系统连接，并实现与同一工作环境中不同制造商的 AGV 和传统系统 (库存系统) 的并行允许。

应定义一个主控与自动导引车 (AGV) 之间的统一接口。这应通过以下要点实现：

- 描述了 AGV 与主控系统之间通信的标准，并以此为基础，实现运输系统融入连续过程自动化，采用协同运输车辆。
- 通过增加车辆自主性、过程模块和接口，以及其他因素，提高灵活性，并最好将事件控制的命令链的刚性序列分离。
- 由于具有高度的"即插即用"能力，实施时间得以减少，所需信息（例如订单信息）由中央服务提供且通常有效。车辆应能独立于制造商进行投入运行，同时考虑职业安全要求，以相同的实施努力。
- 通过对所有运输车辆、车辆型号和制造商使用统一、总体的协调以及相应的逻辑，减少系统复杂性并提高"即插即用"能力。
- 通过车辆控制与协调层之间的通用接口，提高制造商的独立性。
- 通过在专有主控系统和上级主控系统之间实现垂直通信，集成专有 DTS 库存系统（参见图 1）。

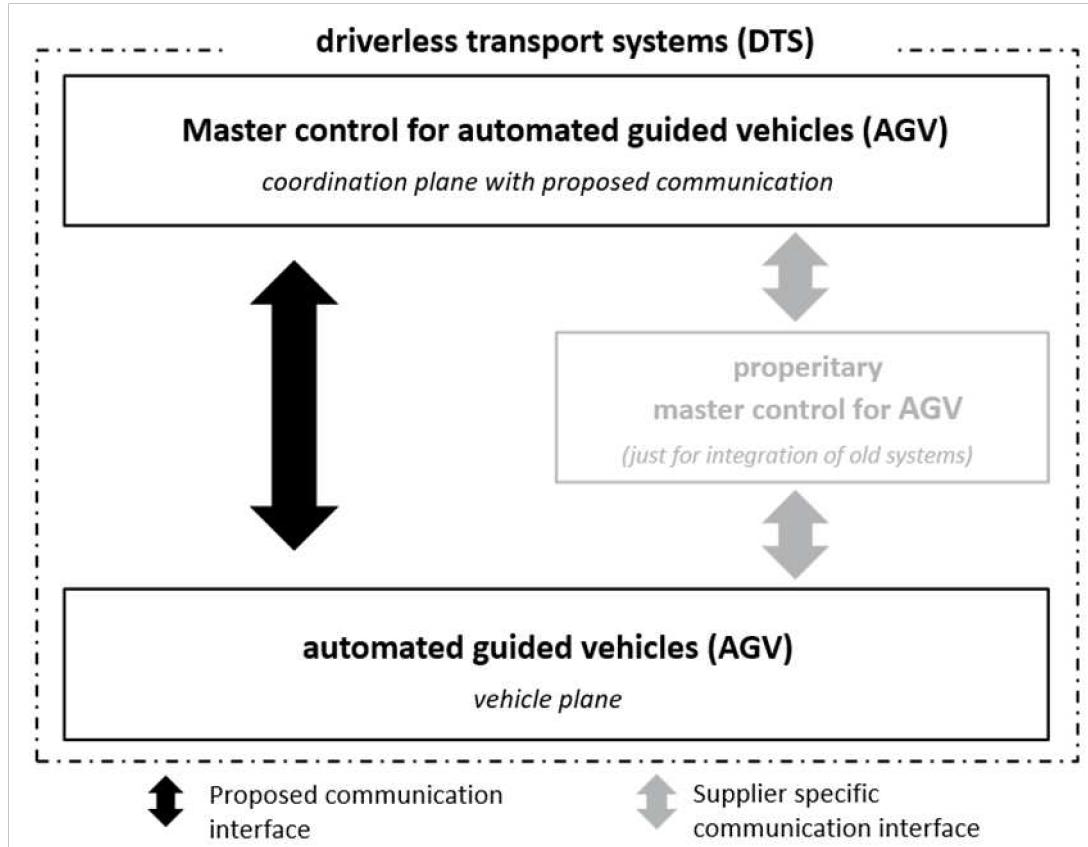


图 1 DTS 库存系统集成

为了实现上述目标，本文档描述了 AGV 与主控之间订单和状态信息通信的接口。

AGV 与主控之间操作所需的接口（例如，在路径规划等方面自由考虑特殊技能等）或与其他系统组件（例如，外部外围设备、防火门等）通信所需的接口，最初不包括在本文档中。

3 范围

本推荐包含有关自动导引车（AGV）与主控之间通信的定义和最佳实践。其目标是允许具有不同特性（例如，牵引车或叉车 AGV）的 AGV 以统一的语言与主控进行通信。这为在主控中操作任何组合的 AGV 奠定了基础。主控提供指令并协调 AGV 的交通。

该接口基于汽车行业生产和工厂物流的需求。根据制定的需求，内部物流的需求涵盖了物流部门的需求，即从货物接收到生产供应再到货物发出的物流过程，通过控制自由导航车辆和引导车辆。

与自动化车辆不同，自主车辆能够在基于相应传感器系统和算法的基础上独立解决出现的问题，并能根据动态环境的变化及时做出反应或进行调整。自主车辆的特性，如独立绕过障碍物，既可以由自由导航车辆实现，也可以由引导车辆实现。然而，一旦路径规划是在车辆本身上进行的，本文件将描述自由导航车辆（参见 8 术语表）。自主系统并非完全去中心化（群体智能），而是通过预定义的规则来定义其行为。

为了实现可持续的解决方案，下面描述了一个可以扩展其结构的接口。这应该能够为被引导车辆的主控提供全面覆盖。自由导航的车辆也可以整合到该结构中；对于这一点所需的详细规范并不包含在本推荐中。

为了整合专有的库存系统，可能需要单独定义接口，这些定义并不包含在本推荐中。

3.1 其他适用文档

Document	Version	Description
VDI Guideline 2510	October 2005	Driverless transport systems (DTS)
VDI Guideline 4451 Sheet 7	October 2005	Compatibility of driverless transport systems (DTS) - DTS master control
DIN EN ISO 3691-4	December 2023	Industrial Trucks Safety Requirements and VerificationPart 4: Driverless trucks and their systems
LIF – Layout Interchange Format	March 2024	Definition of a format of track layouts for exchange between the integrator of the driverless transport vehicles and a (thirdparty) master control system.

4 要求和协议定义

该通信接口旨在支持以下要求：

- 控制最少 1000 辆车辆
- 实现具有不同自主性程度的车辆的集成
- 允许做出决定，例如，关于路线的选择或十字路口的行为

车辆应定期或在状态发生变化时转移其状态。

通信通过无线网络进行，考虑到连接失败和消息丢失的影响。

消息协议为消息队列遥测传输（MQTT），并应与 JSON 结构结合使用。在制定此协议时测试了 MQTT 3.1.1 版本，并且这是为了兼容性所需的最低版本。MQTT 允许将消息分发到子通道，这些子通道称为“主题”。MQTT 网络中的参与者订阅这些主题，并接收与其相关或感兴趣的任何信息。

JSON 结构允许将来在协议中添加额外参数。参数用英文描述，以确保协议具有可读性、易理解性，并适用于德语以外的地区。

5 通信的内容和过程

AGV 的操作至少包括以下参与者：

- AGV 系统操作员提供基本信息
- 主控组织和管理操作
- AGV 执行订单

图 2 描述了操作阶段的通信内容。在实施或修改过程中，AGV 和主控是手动配置的。

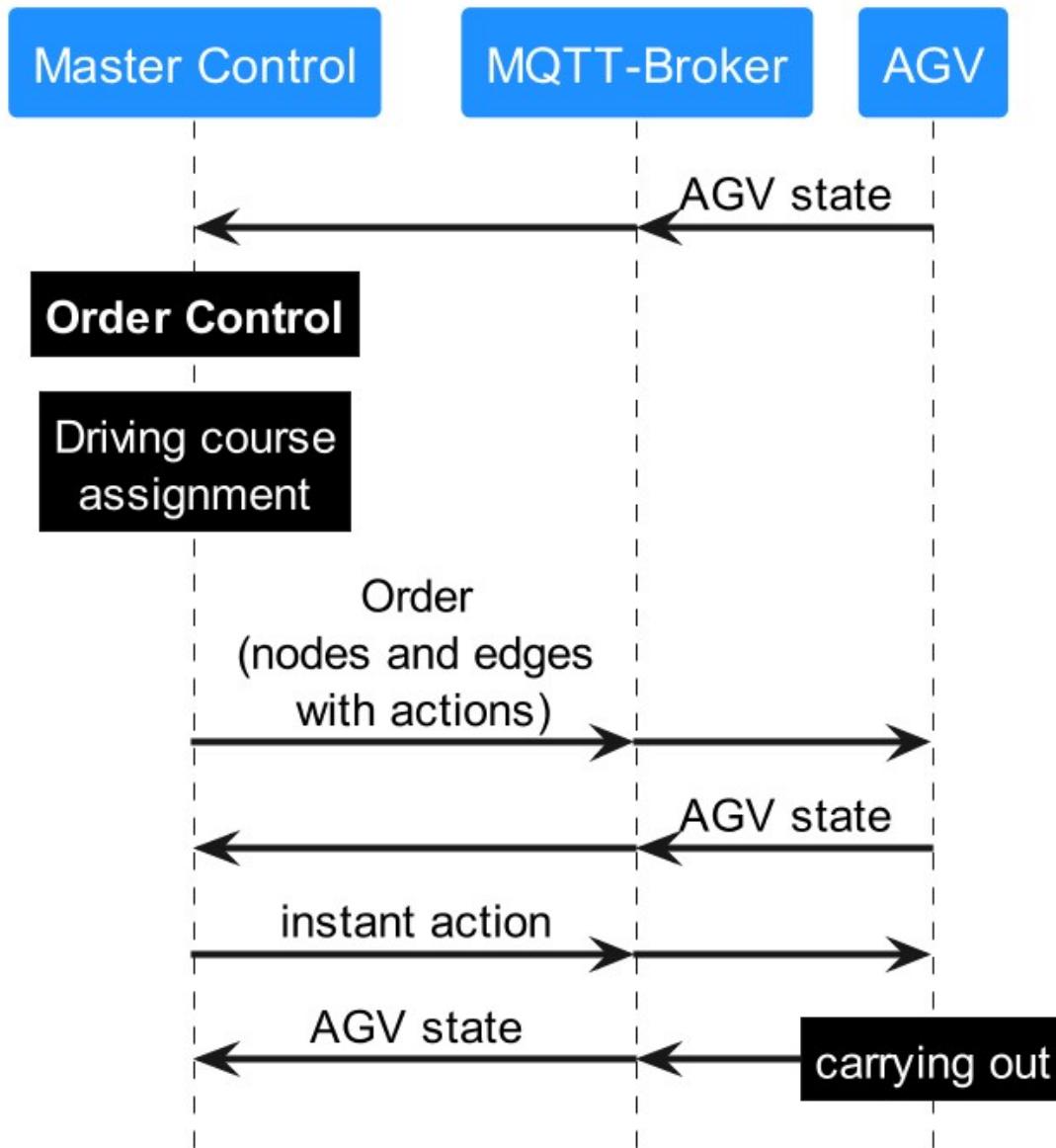


图 2 信息流结构

在实施阶段，由主控和 AGV 组成的无人驾驶运输系统（DTS）将被设置起来。操作方定义必要的框架条件，并将所需信息手动输入或通过从其他系统导入存储到主控中。主要涉及以下内容：

- 路线定义：使用 CAD 导入，可以将路线导入到主控件中。或者，操作员也可以在主控件中手动实现路线。路线可以是单行道，限制某些车辆组（基于尺寸比）等。
- 路线网络配置：在路线中定义了装卸站、电池充电站、周边环境（门、电梯、障碍物）、等待位置、缓冲站等。
- 车辆配置：操作员存储了 AGV 的物理属性（尺寸、可用载具固定装置等）。AGV 应通过第 6.15 主题 "factsheet" 中定义的方式，通过 factsheet 话题进行通信。

上述描述的路线和路线网络配置并不属于本文件。它们为基于这些信息和运输需求，由主控实现订单控制和行驶路线分配提供了基础。然后，通过 MQTT 消息代理将生成的 AGV 订单传输到车辆。在执行任务的同时，AGV 会持续向主控报告其状态。这也通过 MQTT 消息代理完成。

主控功能包括：

- 将订单分配给 AGV
- 计算并引导 AGV 的行驶路线（考虑到每个 AGV 的个体物理特性限制，例如尺寸、机动性等）
- 检测并解决堵塞问题（“死锁”）
- 能源管理：充电指令可以中断运输指令
- 交通控制：缓冲路线和等待位置
- 环境的临时变化，如释放某些区域或改变最大速度
- 与门、闸门、电梯等外围系统通信
- 检测并解决通信错误

AGV 的功能包括：

- 定位
- 沿关联路线导航（引导或自主）
- 执行动作
- 持续传输车辆状态

此外，在配置整个系统时，集成商应考虑以下内容（不完全列表）：

- 地图配置：主控和 AGV 的坐标系统应匹配。
- 枢轴点：使用 AGV 的不同点或充电点作为枢轴点会导致车辆的不同包络。参考点可能因情况而异，例如，承载负载的 AGV 和不承载负载的 AGV 可能有所不同。

6 协议规范

以下部分描述了通信协议的细节。该协议规定了主控与 AGV 之间的通信。AGV 与外围设备之间的通信，例如 AGV 与门之间的通信，不在本协议范围内。

不同的消息在表格中显示，描述了作为订单、状态等发送的 JSON 字段的内容。

此外，验证用的 JSON 模式已在公共 git 仓库 (<https://github.com/VDA5050/VDA5050>) 中提供。每当发布 VDA5050 时，JSON 模式都会更新。如果 JSON 模式与本文件存在差异，则以本文件中的版本为准。

6.1 表格符号及格式含义

该表格包含标识符的名称、其单位、其数据类型，以及（如有）描述。

识别	描述
标准	变量是一个基本数据类型
加粗	变量是一个非基本数据类型（例如 JSON 对象或数组），并单独定义
斜体	变量是可选的
斜体和加粗	变量是可选的，且为非基本数据类型
arrayName[arrayDataType]	变量（此处为 arrayName）是一个数据类型为 arrayDataType 的数组，方括号中包含的内容（此处为数据类型 arrayDataType）

所有关键字区分大小写。所有字段名使用驼峰命名法。所有枚举值使用全大写且不带下划线。

6.1.1 可选字段

如果一个变量被标记为可选，这意味着对于发送方来说是可选的，因为该变量在某些情况下可能不适用（例如，当主控向 AGV 发送指令时，某些 AGV 会自行规划轨迹，指令的 edge 对象内的 trajectory 可以被省略）。

如果 AGV 接收到包含在此协议中标记为可选字段的指令，则 AGV 应相应地处理，不能忽略该字段。如果 AGV 无法相应地处理该指令，则预期的行为是在错误消息中传达此信息并拒绝该指令。

主控应仅发送 AGV 支持的可选信息。

示例：轨迹是可选的。如果 AGV 无法处理轨迹，主控不应向车辆发送轨迹。

AGV 应通过 AGV factsheet 消息来沟通它需要的可选参数。

6.1.2 允许的字符和字段长度

所有通信均使用 UTF-8 编码，以实现描述的国际适配。建议 ID 应仅使用以下字符：

A-Z a-z 0-9 _ - :

最大消息长度未定义，但受 MQTT 协议规范和或许受 factsheet 中定义的技术限制。如果 AGV 的内存不足以处理传入的订单，则应拒绝该订单。最大字段长度、字符串长度或值范围的匹配由集成商决定。为了便于集成，AGV 供应商应提供详细说明 6.15.1 Factsheet JSON 结构 的 AGV Factsheet。

6.1.3 字段、主题和枚举的注解

本文档中的主题和字段以以下方式突出显示：exampleField 和 exampleTopic。枚举应大写。这些值在文档中以单引号括起来。这包括关键字，如 actionStatus ('WAITING'、'FINISHED' 等)。

6.1.4 JSON 数据类型

在可能的情况下，应使用 JSON 数据类型。布尔值因此由"true"或"false"编码，而不是枚举 ('TRUE'、'FALSE') 或魔法数字。数值数据类型通过类型和精度指定，例如 float64 或 uint32。不支持的 IEEE 754 的特殊数值，如 NaN 和 infinity。

6.2 MQTT 连接处理、安全性和 QoS

MQTT 协议提供了为客户端设置最后遗嘱消息的选项。如果客户端因任何原因意外断开连接，代理会向其他订阅的客户端分发最后遗嘱。此功能的使用在 6.14 主题 "connection" 中描述。

如果自动导引车（AGV）与代理服务器断开连接，它会保留所有订单信息，并完成订单直至最后一个已发布的节点。

协议安全性需要在代理配置中考虑。

为了减少通信开销，MQTT QoS 级别 0（Best Effort）将用于主题 order、instantActions、state、factsheet 和 visualization。主题连接应使用 QoS 级别 1（至少一次）。

6.3 MQTT 主题层级

由于云服务提供商强制规定了主题结构，MQTT 主题结构并未严格定义。对于基于云的 MQTT 代理，主题结构必须单独调整以匹配本协议中定义的主题。这意味着以下各节中定义的主题名称是强制性的。

对于本地代理，建议的 MQTT 主题层级如下：

interfaceName/majorVersion/manufacturer/serialNumber/topic

例如：

uagv/v2/KIT/0001/order

MQTT 主题层级	数据类型	描述
interfaceName	string	使用的接口名称
majorVersion	string	VDA 5050 的主版本号 推荐, 以 "v" 开头
manufacturer	string	自动导引车 (AGV) 的制造商。
serialNumber	string	序列号由以下字符组成: A-Z a-z 0-9 - . : -
topic	string	主题 (例如, 订单或状态) 参见第 6.5 通信主题

注意: 由于 / 字符用于定义主题层次结构, 因此不得在任何上述字段中使用。\$字符也用于某些 MQTT 代理的特殊内部主题, 因此也不应使用。

6.4 协议头

每个 JSON 消息以 header 开始。在以下章节中, 为了可读性, 将引用以下字段作为 header。头部由以下独立元素组成。header 不是 JSON 对象。

对象结构/标识符	数据类型	描述
headerId	uint32	消息的标题 ID。每个主题都定义了 headerId, 并且每个发送 (但不一定接收) 的消息都会增加 1。
timestamp	string	时间戳 (ISO 8601, UTC); 格式为: YYYY-MM-DDTHH:mm:ss.ffZ (例如, "2017-04-15T11:40:03.12Z")。
version	string	版本协议 [主版本].[次版本].[补丁版本] (例如, 1.3.2)。
manufacturer	string	AGV 的制造商。
serialNumber	string	AGV 的序列号。

协议版本

协议版本使用语义版本控制作为版本控制方案。

主版本变更示例：

- 破坏性改变，例如新增非可选字段

次版本变更示例：

- 新功能，如可视化示例的附加主题

补丁版本示例：

- 电池充电更高的可用精度

6.5 通信主题

AGV 协议使用以下主题在主控和 AGV 之间进行信息交换。

Topic name 主题名称	Published by 发布者	Subscribed by 订阅者	Used for 用于	Implementation 实现	Schema 验证规则
order	master control	AGV	从主控到 AGV 的行驶订单的沟通	mandatory 强制的	order.schema
instantActions	master control	AGV	将要立即执行的操作的通信	mandatory 强制的	instantActions.schema
state	AGV	master control	AGV 状态的通信	mandatory 强制的	state.schema
visualization	AGV	visualization systems	更高频率的位置主题 仅用于可视化目的	optional 可选的	visualization.schema
connection	Broker/ AGV	master control	指示 AGV 连接丢失的时间，不用于主控制检查车辆运行状况，为连接的 MQTT 协议级别检查添加	mandatory 强制的	connection.schema
factsheet	AGV	master control	参数或供应商特定信息，以协助在主控件中设置 AGV	mandatory 强制的	factsheet.schema

6.6 主题: "order" (从主控到 AGV)

主题 “order” 是 MQTT 主题，AGV 通过该主题接收 JSON 封装 order。

6.6.1 概念和逻辑

订单的基本结构是由节点和边组成的图。AGV 需要遍历节点和边来执行订单。所有连接的节点和边的完整图由主控持有。

主控中的图表示包含限制，例如哪些 AGV 允许遍历哪些边。这些限制不会传达给 AGV。主控仅包含 AGV 允许遍历的边。

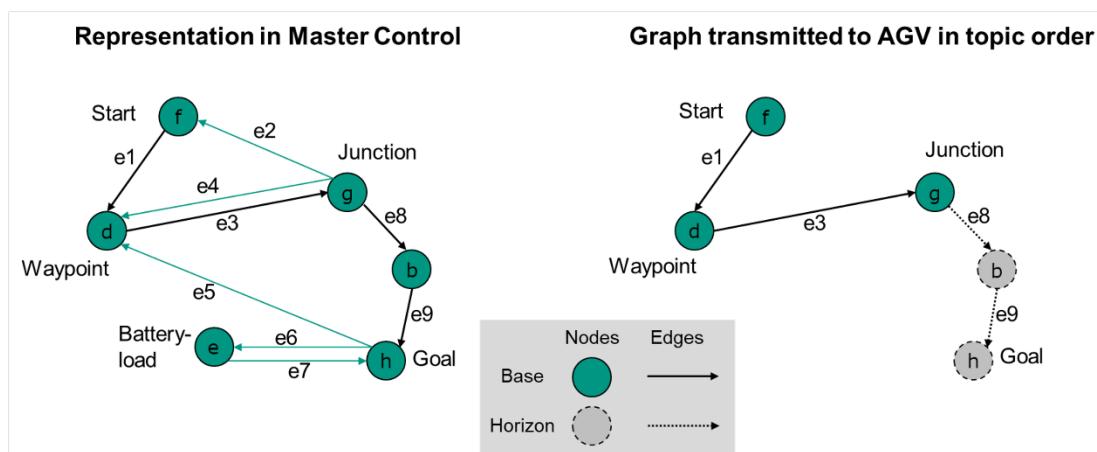


图 3 主控中的图表示和订单中传输的图

节点和边作为两个列表在订单消息中传递。这些列表中节点和边的顺序也决定了节点和边应当以何种顺序被遍历。

对于有效的订单，至少应有一个节点，边的数量应等于节点数量减一。

订单的第一个节点对于 AGV 来说是显然可达的。这意味着 AGV 已经站在该节点上，或者 AGV 位于该节点的偏差范围内。

节点和边都具有一个布尔属性 `released`。如果一个节点或边被释放，则预期 AGV 会穿越它。如果一个节点或边未被释放，则 AGV 不应穿越它。

只有当边的起始节点和终止节点都释放时，该边才能被释放。

在未释放的边之后，序列中不能跟随任何已释放的节点或边。

已释放的节点和边的集合称为“base”，未释放的节点和边的集合称为“horizon”。

可以发送没有 horizon 的订单。

订单消息不一定描述完整的运输订单。为了交通控制和适应资源受限的车辆，完整的运输订单（可能包含许多节点和边）可以被拆分成多个子订单，这些子订单通过它们的 `orderId` 和 `orderUpdateId` 连接。更新订单的过程在下一节中描述。

6.6.2 订单和订单更新

为了支持交通管理，主控制可以将通过订单通信的路径分成两部分：

- “Base”: 这是 AGV 被允许行驶的预定路线。基础路线的所有节点和边都已由主控为车辆释放。基础路线的最后一个节点称为决策点。
- “Horizon”: 这是主控为 AGV 在决策点之后规划的当前路线。horizon 路线尚未由主控释放。

如果基础路线不再添加更多节点和边，AGV 应在决策点停止。为确保流畅移动，如果交通情况允许，主控系统应在 AGV 到达决策点之前扩展基础路线。

由于 MQTT 是一种异步协议，且无线网络传输不可靠，基础路线不能更改。因此，主控系统应假定 AGV 已经执行了基础路线。后文将描述取消订单的流程，但由于上述通信限制，这也被视为不可靠。

主控系统有可能通过向 AGV 发送更新的路线来改变 horizon，其中包括更改的节点和边的列表。改变 horizon 路线的过程如图 4 所示。

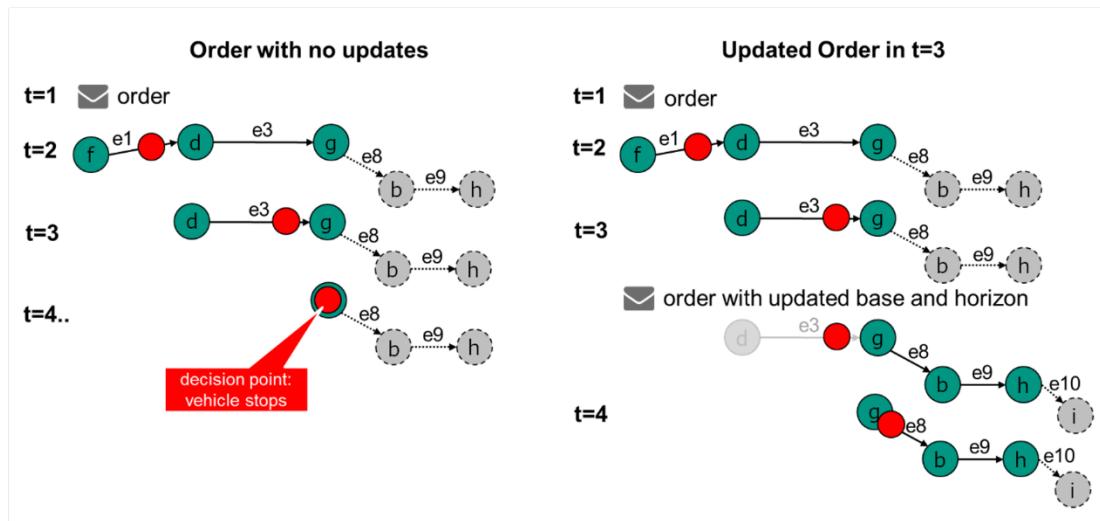


图 4 变更行驶路线“Horizon”的流程

在图 4 中，控制面板在时间 t=1 时首先发送一个初始任务。图 5 展示了任务的伪代码。为了提高可读性，这里省略了一个完整的 JSON 示例。

```
{
  orderId: "1234",
  orderUpdateId: 0,
  nodes: [
    f {released: True},
    d {released: True},
    g {released: True},
    b {released: False},
    h {released: False}
  ],
  edges: [
    e1 {released: True},
    e3 {released: True},
    e8 {released: False},
    e9 {released: False}
  ]
}
```

图 5 订单的伪代码

在时间 $t = 3$, 通过发送订单扩展来更新订单 (见**图 6** 中的示例)。请注意, `orderUpdateId` 会增加, 并且订单更新的第一个节点对应于先前订单消息的最后一个共享基础节点。

这确保了 AGV 也能执行工作更新, 即订单更新的第一个节点可以通过执行 AGV 已知的边来访问。

```
{
  orderId: 1234,
  orderUpdateId: 1,
  nodes: [
    g {released: True},
    b {released: True},
    h {released: True},
    i {released: False}
  ],
  edges: [
    e8 {released: True},
    e9 {released: True},
    e10 {released: False}
  ]
}
```

图 6 订单更新的伪代码。注意 `orderUpdateId` 的变化

这也有助于在订单更新丢失的情况下 (例如, 由于不可靠的无线网络)。AGV 可以始终检查最后已知的基础节点是否具有相同的 `nodeId` (以及 `nodeSequenceId`, 稍后会详细说明) 作为第一个新的基础节点。

也请注意, 节点 `g` 是唯一一个被重新发送的基节点。由于基节点不能更改, 因此节点 `f` 和 `d` 的重新传输是无效的。

确保拼接节点的（示例中为节点 g）内容不被更改非常重要。对于操作、偏差范围等，AGV 应使用第一指令（图 5, orderUpdateId 0）中提供的指令。

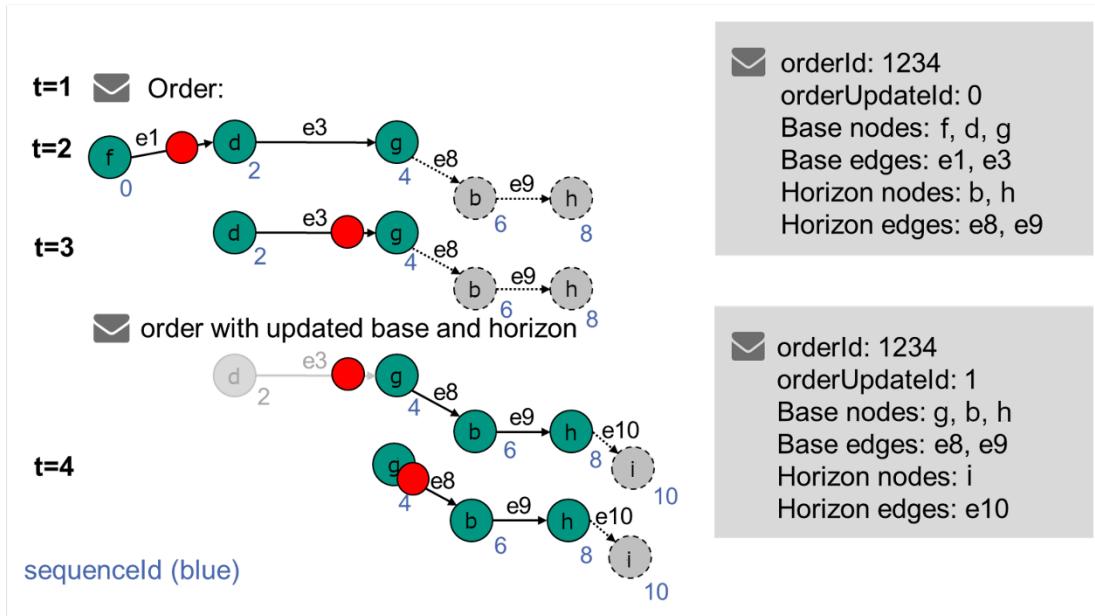


图 7 定期更新流程 - 订单扩展

图 7 描述了如何扩展订单。它显示了 AGV 当前可用的信息。orderId 保持不变，而 orderUpdateId 会递增。

上一个基础节点的最后一个节点是更新顺序中的第一个基础节点。通过这个节点，AGV 可以将更新后的订单添加到当前订单（拼接）。来自上一个基础的其他节点和边不会再次发送。

主控可以选择通过发送完全不同的节点作为新基础来修改 horizon。horizon 也可以被删除。

为了允许订单中的循环（例如从节点 a 到 b 然后返回 a），节点和边对象会被分配一个 sequenceld。这个 sequenceld 会遍历节点和边（订单的第一个节点获得 0，第一个边获得 1，第二个节点获得 2，以此类推）。这有助于更轻松地跟踪订单的进度。

一旦分配了 sequenceld，它不会随着顺序更新而改变（见 图7）。这是为了在 AGV 端确定主控指向哪个节点所必需的。

图 8 描述了接受订单或订单更新的流程。

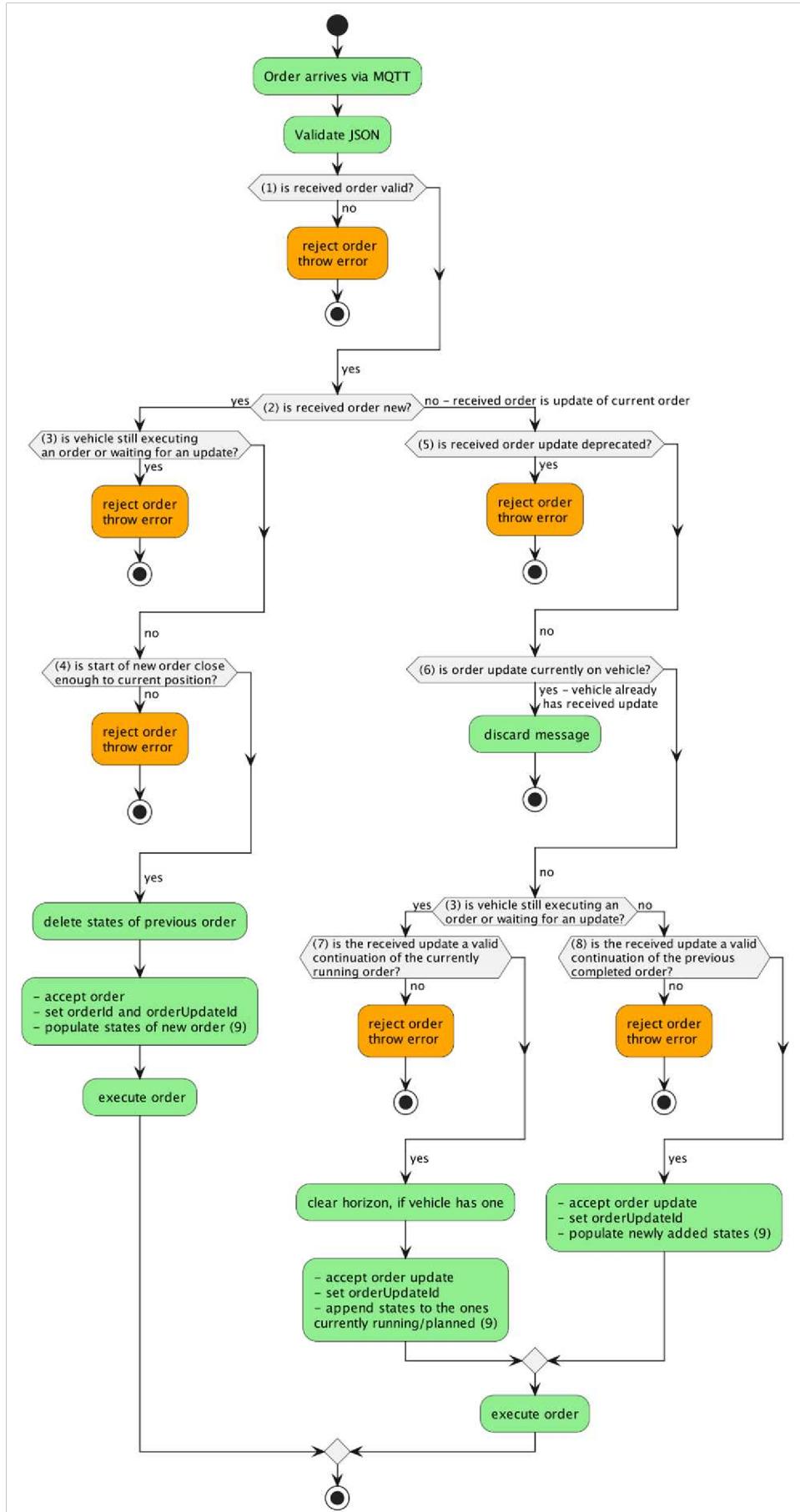


图 8 接收订单或订单更新的过程

1. **is received order valid?**: 所有格式和 JSON 数据类型是否正确?
2. **is received order new or an update of the current order?**: 收到的订单的 orderId 是否与车辆当前持有的订单的 orderId 不同?
3. **is vehicle still executing an order or waiting for an update?**: nodeStates 不是空的, 还是 actionStates 包含既不是 “FAILED” 也不是 “FINISHED”的状态? 节点和边以及订单 horizon 的相应动作状态也包含在状态中。车辆可能仍有 horizon, 因此等待更新并执行订单。
4. **is start of new order close enough to current position?**: 车辆是否已经在节点, 或者是否在节点的偏差范围内(参见 6.6.1 概念和逻辑)?
5. **is received order update deprecated?**: orderUpdateId 是否小于车辆上当前的值?
6. **is received order update currently on vehicle?**: orderUpdateId 是否等于车辆上当前的值?
7. **is the received update a valid continuation of the currently still running order?**: 收到的任务更新的第一个节点是否等于当前决策点 (当前基准的最后一个节点) ? 车辆仍在移动或执行先前任务更新中释放的与基准相关的操作, 或者仍有 horizon 并因此等待任务的延续。在这种情况下, 任务更新只有在新基准的第一个节点等于先前基准的最后一个节点时才会被接受。
8. **is the received update a valid continuation of the previously completed order?**: 收到的任务更新中第一个节点的 nodeId 和 sequenceId 是否等于 lastNodeId 和 lastNodeSequenceId? 车辆不再执行任何操作, 也不再等待任务的延续 (这意味着它已经完成了所有相关操作并没有了视野范围) 。任务更新现在会被接受, 因为它从最后一个经过的节点继续, 因此新基准的第一个节点需要与车辆的 lastNodeId 以及 lastNodeSequenceId 相匹配。
9. 填充/追加状态指的是 actionStates / nodeStates / edgeStates。

6.6.3 订单取消 (由主控发起)

在基础节点发生非计划性变化的情况下, 应使用 instantAction cancelOrder 来取消订单。

收到 instantAction cancelOrder 后, 车辆停止 (根据其功能, 例如停留在原地或下一个节点) 。

如果有预定的操作, 这些操作应被取消, 并在 actionState 中报告 “FAILED” 。如果有正在运行的操作, 这些操作应该被取消, 并报告为 “FAILED” 。如果操作不能被中断, 该操作的 actionState 应该通过在运行时报告 “RUNNING” 来反映这一点, 然后报告各自的状态 (如果成功, 则报告 “FINISHED”, 如果失败, 则报告 “FAILED”) 。当操作正在运行时, cancelOrder 操作将报告 “running”, 直到所有操作被取消/完成。在车辆的所有运动及其所有动作停止后, cancelOrder 动作状态将报告为 “FINISHED”。

orderId 和 orderUpdateId 保持不变。

图 9 显示了不同 AGV 能力下的预期行为。

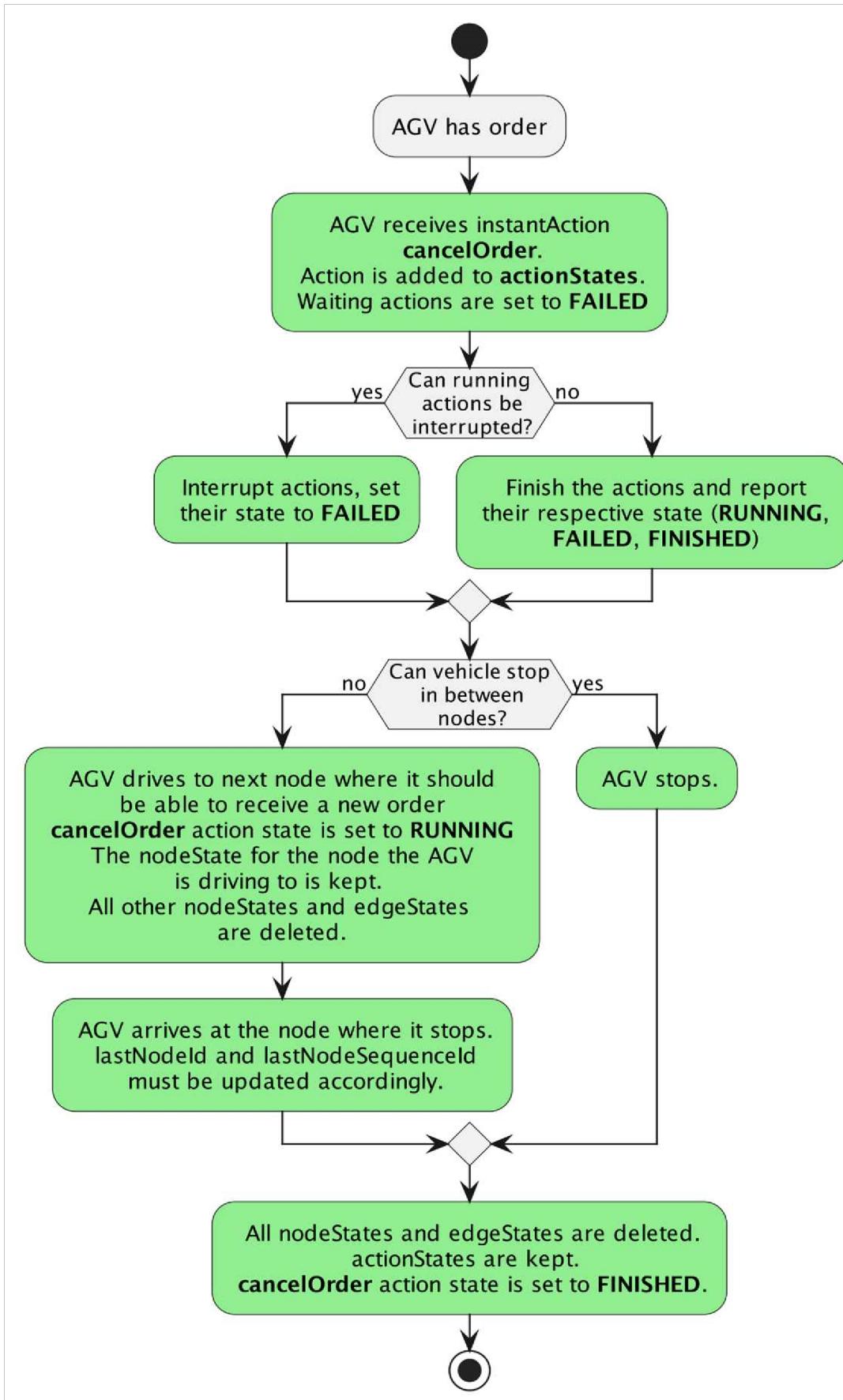


图 9 取消订单后的预期行为

6.6.3.1 取消订单后接收新订单

订单取消后，车辆应处于接收新订单的状态。对于通过标签在节点上定位的 AGV，新订单必须从 AGV 当前所在的节点开始（另见图 5）。

对于可以在节点之间停泊的 AGV，如何开始下一个订单的选择由主控决定。AGV 应接受这两种方法。

两种选项：

- 发送一个指令，其中第一个节点是一个临时节点，该节点位于 AGV 当前所在的位置。AGV 应认识到该节点可以轻易到达并接受该指令。
- 发送一个指令，其中第一个节点是上一个指令中最后经过的节点，但设置偏差范围足够大，使 AGV 位于该范围内。这样，AGV 应认识到该节点应被计为已经过并接受该指令。

6.6.3.2 当 AGV 没有订单时接收 cancelOrder 操作

如果 AGV 接收到 cancelOrder 操作，但 AGV 当前没有订单，或者上一个订单已被取消，则 cancelOrder 操作应报告为'FAILED'。

AGV 应报告一个“noOrderToCancel”的错误，并将 errorLevel 设置为'WARNING'。

instantAction 的 actionId 应作为错误引用传递。

6.6.4 拒绝订单

存在几种情况下订单应被拒绝。这些情况如图 8 所示，并描述如下。

6.6.4.1 车辆收到格式错误的新订单

解决方案：

1. 车辆不会在其内部缓冲区中接管新订单。
2. 车辆报告警告信息"validationError"
3. 警告应持续报告，直到车辆接受新订单为止。

6.6.4.2 车辆收到包含其无法执行的操作或无法使用的字段的订单

示例：

- 不可执行动作：提升高度高于最大提升高度，即使没有安装行程，提升动作等。
- 不可用字段：trajectory 等

解决方案：

1. 车辆不接管其内部缓冲区中的新订单
2. 车辆报告带有错误字段的警告 “orderError” 作为错误参考
3. 警告应持续报告，直到车辆接受新订单为止

6.6.4.3 车辆接收到一个具有相同 orderId 但 orderUpdateId 低于当 前 orderUpdateId 的新订单

解决方案:

1. 车辆不在其内部缓冲区中接管新订单。
2. 车辆在其缓冲区中保留之前的订单。
3. 车辆报告警告"orderUpdateError"
4. 车辆继续执行之前的订单。

如果 AGV 接收到具有相同 orderId 和 orderUpdateId 的订单两次，第二次订单将被忽略。这可能会发生，如果主控重新发送订单，因为状态消息被主控接收得太晚，因此它无法验证第一个订单是否已被接收。

6.6.5 走廊

可选的 `corridor` 边缘属性允许车辆偏离边缘轨迹以避障，并定义允许车辆运行的边界。

要使用 `corridor` 属性，需要一个预定义的轨迹，如果未定义 `corridor` 属性，车辆将遵循该轨迹。这可以是主控制器已知的在车辆上定义的轨迹，也可以是按 `order` 发送的轨迹。使用 `corridor` 属性的车辆的行为仍然是线路引导车辆的行为，只不过它允许暂时偏离轨迹以避开障碍物。

备注：`order` 内的边定义两个节点之间的逻辑连接，而不一定是车辆从起始节点行驶到结束节点时遵循的（真实）轨迹。根据车辆类型，车辆在起始节点和结束节点之间采取的轨迹要么由主控制通过轨迹边缘属性定义，要么作为预定义轨迹分配给车辆。根据车辆的内部状态，选择的轨迹可能会有所不同。

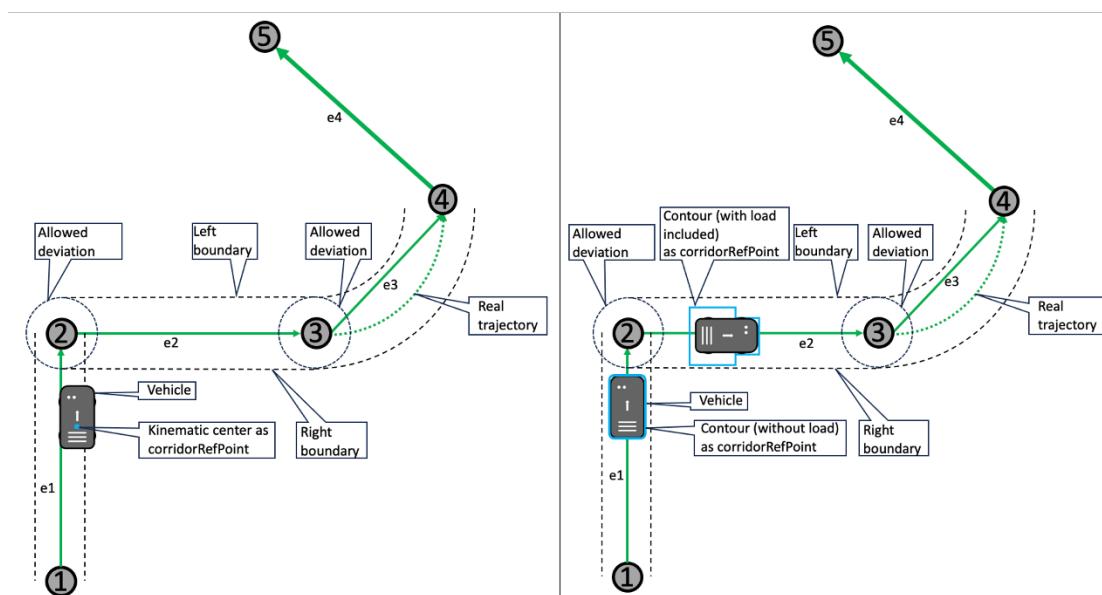


图 10 具有 `corridor` 属性的边缘，定义了允许车辆偏离其预定义轨迹以避开障碍物的左右边界。左侧，运动中心定义了允许的偏差，而右侧，车辆的轮廓（可能因负载而延伸）定义了允许的偏差。这是由 `corridorRefPoint` 参数定
义的。

允许车辆独立导航（并偏离原始边缘轨迹）的区域由左边界和右边界定义。可选的 **corridorRefPoint** 字段指定车辆控制点或车辆轮廓是否应位于定义的边界内。边缘的边界应以这样的方式定义：车辆一经过节点就位于新的当前边缘的边界内。如果车辆不偏离轨迹，主控不应使用 **corridor** 属性，而不是将走廊边界设置为零。

车辆的运动控制软件应不断检查车辆是否在定义的边界内。否则，车辆应因超出允许的导航空间而停车并报告错误。主控制器可以通过取消当前命令并向车辆发送带有允许车辆再次移动的走廊信息的新命令来决定是否需要用户交互或者车辆是否可以继续。

备注：允许车辆偏离轨迹会增加车辆在行驶过程中可能的足迹。在初始运行时，如果主控根据车辆的足迹做出交通控制决策，则应考虑这种情况。

另请参阅 6.10.2 节点遍历及进入/离开边，触发动作以获取更多信息。

6.6.6 订单消息的实现

对象结构	Unit	数据类型	描述
headerId		uint32	头部 ID 根据主题定义，每次发送（但不一定接收）的消息都会递增 1。
timestamp		string	时间戳 (ISO 8601, UTC); YYYY-MM-DDTHH:mm:ss.ffZ (例如, "2017-04-15T11:40:03.12Z")
version		string	协议版本 [主版本].[次版本].[补丁版本] (e.g., 1.3.2)
manufacturer		string	AGV 的供应商
serialNumber		string	AGV 的序列号
orderId		string	用于识别属于同一订单的多个订单消息。
orderUpdateId		uint32	订单更新标识 每个订单 ID 都是唯一的。 如果订单更新被拒绝，此字段应包含在拒绝消息中。
zoneSetId		string	区域集的唯一标识符，AGV 必须使用该标识符进行导航，或者主控系统使用该标识符进行规划。 可选：有些主控系统不使用分区。 某些 AGV 不理解区域。如果没有使用分区，则不要添加到信息中。
nodes [node]		array	为完成订单而要遍历的节点对象数组。 对于有效订单来说，一个节点就足够了。 在这种情况下，边缘数组留空。

Object structure	Unit	Data type	Description
edges [edge]		array	为完成订单而要遍历的边缘对象数组。 对于有效订单来说，一个节点就足够了。 在这种情况下，边缘数组留空。

Object structure	Unit	Data type	Description
node {		JSON object	
nodeId		string	唯一的节点标识
sequenceId		uint32	编号，用于跟踪顺序中节点和边的顺序，并简化顺序更新。 主要目的是区分在一个 orderId 中传递多次的节点。 变量 sequenceId 贯穿同一订单的所有节点和边，并在发出新的 orderId 时重置。
nodeDescription		string	节点的附加信息
released		boolean	"true" 表示节点是基础的一部分。 "false" 表示节点是 horizon 的一部分。
nodePosition		JSON object	节点位置 对于不需要节点位置的车类型（例如，线引导车辆），此为可选。
actions [action] }		array	要在节点上执行的操作数组。 如果不需要操作，则为空数组。

Object structure	Unit	Data type	Description
nodePosition {		JSON object	定义全局项目特定世界坐标系中地图上的位置。 每层楼都有自己的地图。所有地图都应使用相同的项目特定全局原点。
x	m	float64	参考地图坐标系的地图上的 X 位置。 精确度取决于具体的实施。
y	m	float64	参考地图坐标系的地图上的 Y 位置。 精确度取决于具体的实现。
theta	rad	float64	范围: [-Pi ... Pi] AGV 在节点上的绝对方向。 可选：车辆可以自行规划路径。 如果已定义，AGV 必须在此节点上承担 theta 角度。

Object structure	Unit	Data type	Description
			如果上一个边缘不允许旋转, AGV 应在节点上旋转。如果以下边定义了不同的方向, 但不允许旋转, 则 AGV 在进入边之前在节点上旋转到所需的旋转边。
<i>allowedDeviationXY</i>	m	float64	表示 AGV 应如何精确匹配节点的位置, 才能被视为已通过。。 如果 = 0.0: 不允许偏差 (偏差意味着在 AGV 制造商的正常公差范围内)。 如果 > 0.0: 允许的偏差半径 (单位: 米)。 如果自动导引车 (AGV) 在偏差半径内通过某个节点, 则该节点可以被视为已通过。
<i>allowedDeviationTheta</i>	rad	float64	范围: [0.0 ... Pi] 表示 AGV 在节点上需要达到 theta 定义的方向的精确度。 最低可接受角度是 theta - allowedDeviationTheta, 最高可接受角度是 theta + allowedDeviationTheta。
mapId		string	参考位置所依据的地图的唯一标识。每个地图都有相同特定的全局坐标系原点。当一个 AGV 使用电梯时, 例如, 从出发楼层到目标楼层, 它将消失在出发楼层的地图上, 并在目标楼层的地图上相关电梯节点处重生。
<i>mapDescription</i> }		string	地图附加信息

Object structure	Unit	Data type	Description
action {		JSON object	描述 AGV 可以执行的操作。
<i>actionType</i>		string	"Actions and Parameters"表中第一列所描述的动作名称。定义操作函数。
<i>actionId</i>		string	用于识别动作并将它们映射到状态中的 actionState 的唯一 ID 字符串。 建议: 使用 UUIDs.
<i>actionDescription</i>		string	关于操作的附加信息
<i>blockingType</i>		string	枚举值 {'NONE', 'SOFT', 'HARD'}: 'NONE': 允许行驶和其他动作; 'SOFT': 允许其他动作但不允许行驶;

Object structure	Unit	Data type	Description
			'HARD':在该时间点只允许次动作
actionParameters [actionParameter] }		array	指定动作的 actionParameter 对象数组，例如: "deviceId"、"loadId"、"external triggers"。 一个示例实现可以在 7.2 参数格式找到

Object structure	Unit	Data type	Description
edge {		JSON object	两个节点之间的定向连接。
edgeld		string	边的唯一标识
sequenceld		uint32	编号以跟踪订单中节点和边的顺序，并简化订单更新。 变量 sequenceld 跨越同一订单的所有节点和边，并在发出新的 orderId 时重置。
edgeDescription		string	边的附加信息
released		boolean	"true" 表示该边是 base 的一部分。 "false" 表示该边是 horizon 的一部分。
startNodeld		string	订单中第一个节点的 id。
endNodeld		string	订单中最后一个节点的 id
maxSpeed	m/s	float64	边上允许的最大速度。速度由车辆最快的测量值定义。
maxHeight	m	float64	车辆在边上允许的最大高度，包括负载。
minHeight	m	float64	允许的负载处理设备边最小高度。
orientation	rad	float64	AGV 在边的朝向。值 orientationType 定义了它是相对于全局项目特定地图坐标系进行解释，还是相对于边切向解释。如果解释为相对于边切向，0.0 表示向前行驶，PI 表示向后行驶。 示例：方向为 $\pi / 2$ 弧度将导致 90 度的旋转。 如果 AGV 以不同的方向启动，将车辆在边缘旋转到所需方向，如果 rotationAllowed 设置为 "true" 如果 rotationAllowed 是 "false"，在进入边前旋转。如果不可能，则拒绝该订单。

Object structure	Unit	Data type	Description
			<p>如果没有定义轨迹，将旋转应用于边两个连接节点之间的直线路径。</p> <p>如果为边定义了轨迹，将方向应用于轨迹。</p>
<i>orientationType</i>		string	<p>枚举值{'GLOBAL', 'TANGENTIAL'}:</p> <p>'GLOBAL': 相对于全局项目特定地图坐标系;</p> <p>'TANGENTIAL': 沿边切线方向</p> <p>如果没有定义，默认值为 'TANGENTIAL'.</p>
<i>direction</i>		string	<p>为线引导或线引导车辆在交叉路口设置方向，初始时需定义（按车辆个体）。</p> <p>示例: "left", "right", "straight".</p>
<i>rotationAllowed</i>		boolean	<p>"true": 允许在边上旋转</p> <p>"false": 不允许在边上旋转</p> <p>可选: 如果没有设置，则无限制</p>
<i>maxRotationSpeed</i>	rad/s	float64	<p>最大的旋转速度</p> <p>可选项: 如果没有设置，则无限制</p>
<i>trajectory</i>		JSON object	<p>这个边的轨迹 JSON 对象作为 NURBS。定义 AGV 在边的起始节点和终止节点之间移动的路径。</p> <p>可选:</p> <p>如果 AGV 无法处理轨迹或 AGV 自行规划轨迹，可以省略。</p>
<i>length</i>	m	float64	<p>从起始节点到终止节点的路径长度</p> <p>可选:</p> <p>此值由线引导式自动导引车（AGV）在到达停止位置前使用，以降低其速度。</p>
<i>corridor</i>		JSON object	定义车辆可以偏离其轨迹的边界，例如，为了避开障碍物。
action [action] }		array	<p>边缘上要执行的动作用数组表示。</p> <p>如果不需要任何动作，则为空数组。由边触发的动作仅在 AGV 穿越触发该动作的边时有效。当 AGV 离开该边时，该动作将停止，并恢复进入边前的状态。</p>

Object structure	Unit	Data type	Description
			will stop and the state before entering the edge will be restored.
trajectory {		JSON object	
degree		float64	范围: [1.0 ... float64.max] 定义轨迹的 NURBS 曲线的程度。 如果没有定义, 则默认值为 1.
knotVector [float64]		array	范围: [0.0 ... 1.0] NURBS 的节点值数组。 knotVector 的大小为控制点数量 + degree + 1.
controlPoints [controlPoint] }		array	定义 NURBS 控制点的控制点对象数组, 明确包括起点和终点。

Object structure	Unit	Data type	Description
controlPoint {		JSON object	
x		float64	参考地图坐标系的地图上的 X 位置。
y		float64	参考地图坐标系的地图上的 Y 位置。
weight		float64	范围: [0.0 ... float64.max] 曲线控制点的权重。 当没有定义时,默认值为 1.0.
}			

Object structure	Unit	Data type	Description
corridor {		JSON object	
leftWidth	m	float64	范围: [0.0 ... float64.max] 定义车辆轨迹左侧的 corridor 宽度 (米) (见图 13)。
rightWidth	m	float64	范围: [0.0 ... float64.max] 定义车辆轨迹右侧的 corridor 宽度 (米) (见图 13)。
corridorRefPoint }		string	定义边界对车辆的运动中心还是轮廓有效。 如果未指定, 边界对车辆运动学中心有效。 枚举值{'KINEMATICCENTER', 'CONTOUR'}

6.7 Maps

为确保不同类型 AGV 之间导航的一致性，位置始终参照项目特定的坐标系指定（见图 11）。为了区分场地或位置的不同级别，使用唯一的 mapId。地图坐标系应指定为右手坐标系，其中 z 轴指向天空。因此，正旋转应理解为逆时针旋转。车辆坐标系也指定为右手坐标系，其中 x 轴指向车辆前进方向，z 轴指向上方。车辆参考点是定义为车辆参考系中的(0,0,0)，除非另有指定。这符合 DIN ISO 8855 第 2.11 节的规定。

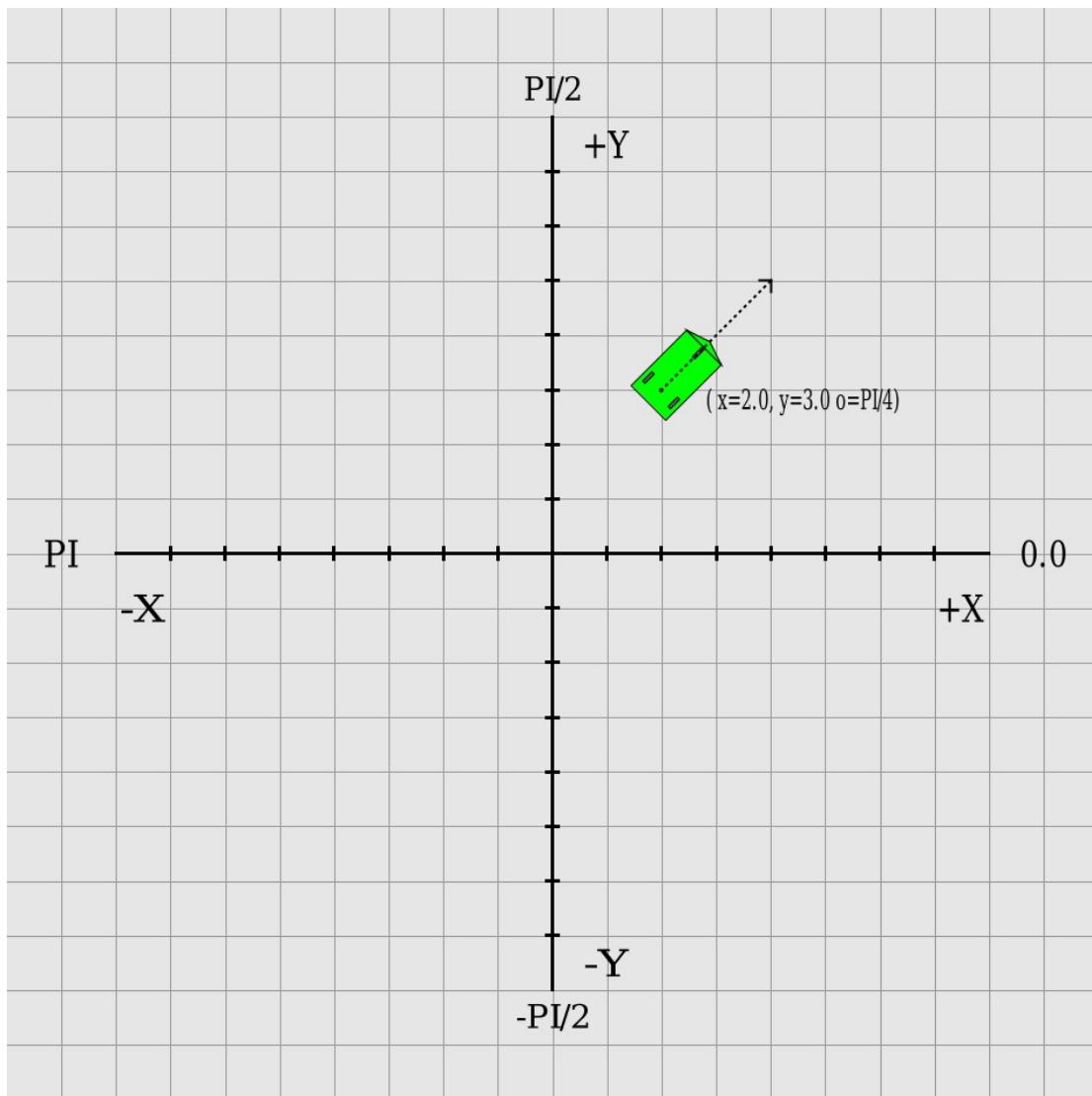


图 11 坐标系示例及 AGV 方向

X、Y 和 Z 坐标应以米为单位给出。方向应以弧度表示，且应在 $+\text{Pi}$ 和 $-\text{Pi}$ 之间。

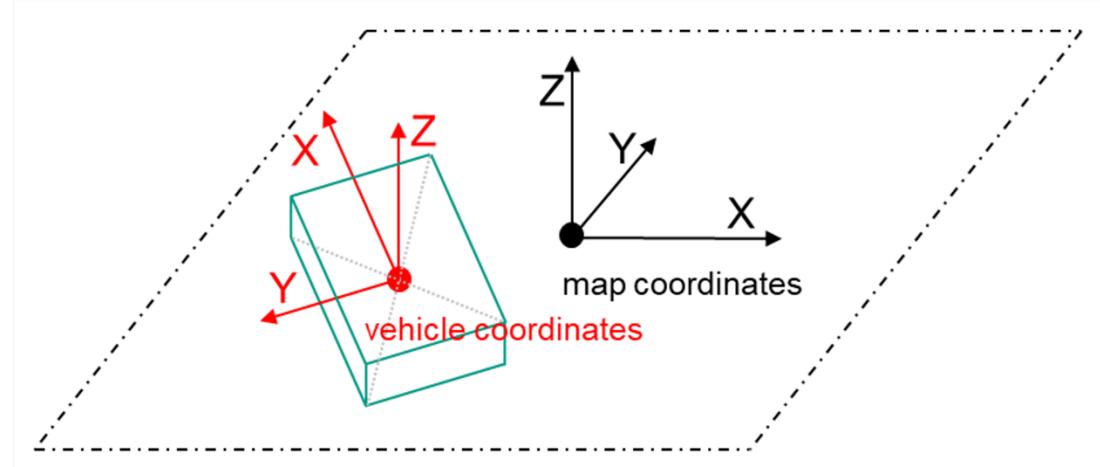


图 12 地图和车辆的坐标系

6.7.1 地图分发

为了实现自动地图分发并在必要时智能管理车辆重启，引入了一种标准化的地图分发方式。

待分发的地图文件存储在车辆可访问的专用地图服务器上。为确保高效传输，每次传输应包含单个文件。如果需要多个地图或文件，应将它们捆绑或打包成一个文件。从地图服务器向车辆传输地图的过程是一个拉取操作，由主控通过使用 instantAction 发出下载命令来启动。

每个地图由地图标识符（字段 mapId）和地图版本（字段 mapVersion）的组合唯一标识。地图标识符描述了车辆物理工作空间的特定区域，地图版本指示对先前版本的更新。在接受新订单之前，车辆应检查请求订单中每个地图标识符的车辆上是否有地图。确保正确激活地图以操作车辆是主控的责任。

为了最小化停机时间并使主控更容易同步新地图的激活，车辆上必须预加载或缓存地图。车辆上地图的状态可以通过车辆状态通道访问。需要注意的是，将地图传输到 AGV 和激活地图是两个不同的过程。要激活车辆上预加载的地图，主控发送一个即时动作。在这种情况下，任何具有相同地图标识符但不同地图版本的地图都会被自动禁用。主控可以通过另一个即时动作删除地图。该过程的结果显示在车辆状态中。

地图分发过程如图 13 所示。

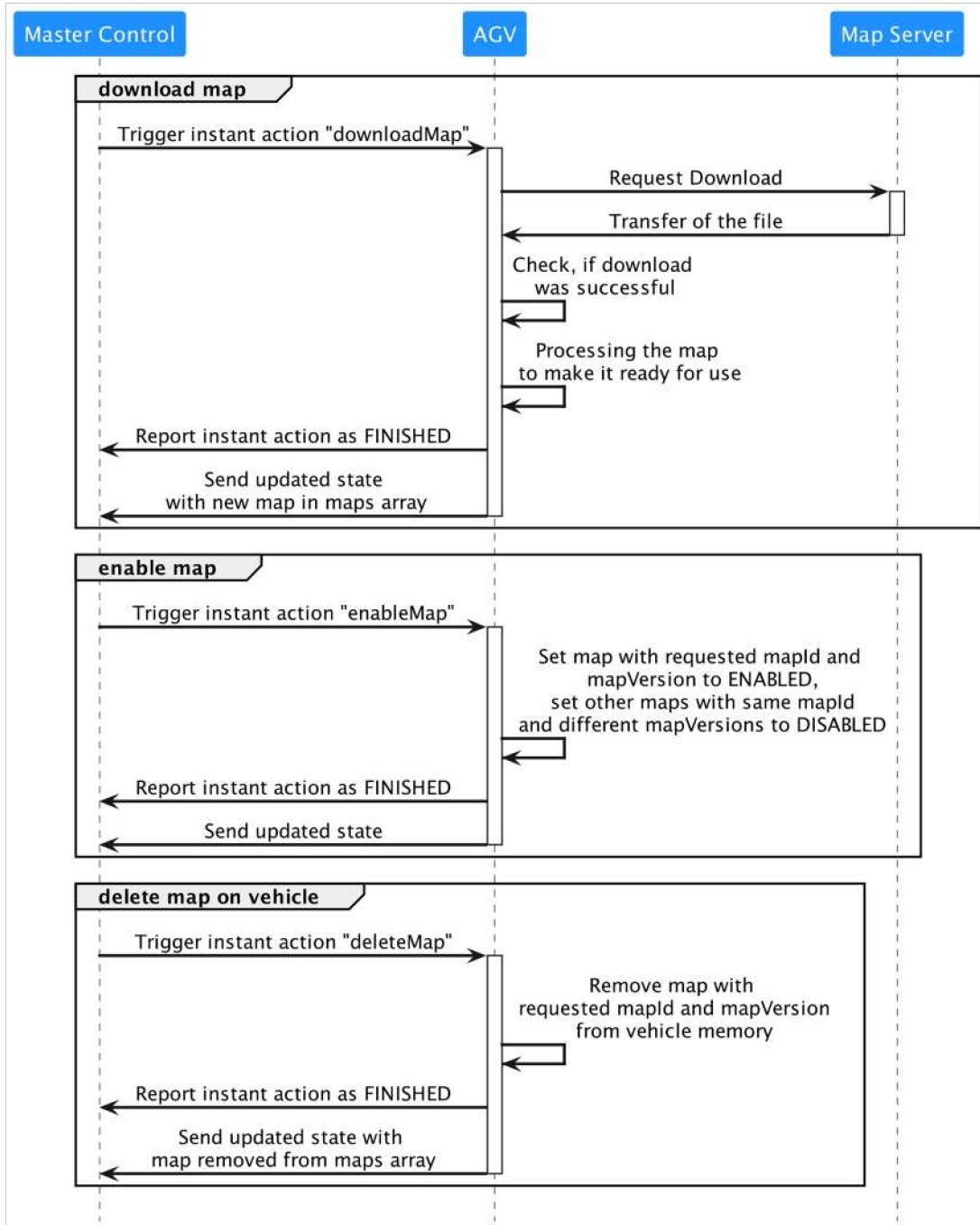


图 13 主控、AGV 和地图服务器之间下载、启用和删除地图所需的通信。

6.7.2 车辆状态中的地图

状态中的 `agvPosition` 字段中的 `mapId` 表示当前激活的地图。车辆上可用的地图信息显示在状态消息的 `maps` 数组中，该数组是状态消息的一个组件。此数组中的每个条目都是一个包含必填字段 `mapId`、`mapVersion` 和 `mapStatus` 的 JSON 对象，这些字段可以是'ENABLED'或'DISABLED'。如果需要，车辆可以使用'ENABLED'的地图。'DISABLED'的地图不应使用。下载过程的当前状态由当前操作未完成表示。状态中也报告错误。

请注意，可以同时启用具有不同 `mapId` 的多个地图。同一时间只能启用一个版本的具有相同 `mapId` 的地图。如果 `maps` 数组为空，这意味着车辆当前没有可用的地图。

6.7.3 地图下载

地图下载由主控的 `downloadMap` 即时操作触发。此命令包含必填参数 `mapId` 和 `mapDownloadLink`，地图存储在地图服务器上，车辆可以访问该链接。

AGV 在开始下载地图文件时将 `actionStatus` 设置为'RUNNING'。如果下载成功，`actionStatus` 将更新为'FINISHED'。如果下载失败，状态将设置为'FAILED'。一旦下载成功完成，地图应当被添加到状态中的地图数组。在地图准备启用之前，不应在状态中报告地图。

确保下载地图的过程不会修改、删除、启用或禁用车辆上现有的任何地图非常重要。车辆应当拒绝下载已在车辆上的具有特定 `mapId` 和 `mapVersion` 的地图。应报告错误，并将即时操作的状态设置为'FAILED'。主控应当首先删除车辆上的地图，然后重新开始下载。

6.7.4 启用已下载的地图

在车辆上启用地图有两种方法：

1. **Master control enables map:** 使用 `enableMap` 即时操作将地图设置为车辆上的'ENABLED'。具有相同 `mapId` 但不同 `mapVersion` 的其他版本将设置为'DISABLED'。
2. **Manually enable a map on the vehicle:** 在某些情况下，可能需要直接在车辆上启用地图。结果应当在车辆状态中报告。

在发送订单中的节点位置时，主控负责确保在车辆上激活正确的地图，同时将相应的 `mapId` 作为一部分发送。如果需要将车辆设置到新地图上的特定位置，则使用 `initPosition` 即时动作。

6.7.5 删除车辆上的地图

主控可以请求从车辆上删除特定地图。这通过即时动作 `deleteMap` 完成。当车辆内存不足时，应向主控报告，主控随后可以启动地图删除操作。车辆本身不允许删除地图。成功删除地图后，必须从车辆状态中的地图数组中移除该地图的条目。

6.8 动作

| 如果 AGV 支持除行驶以外的其他动作，这些动作通过附加到节点或边上的操作字段执行，或通过单独的主题 instantActions 发送（参见第 6.9 节主题：“instantActions”（从主控到控制到 AGV））。

在边上要执行的动作只有在 AGV 位于边上时才能运行（参见第 6.10.2 节节点遍历和进入/离开边缘、触发操作）。

在节点上触发的动作可以一直运行，直到它们需要停止。节点上的操作应该是自终止的（例如，持续五秒的音频信号或取货操作，取货后完成），或者应该成对制定（例如， activateWarningLights 和 deactivateWarningLights），尽管可能会有例外。

下一节介绍了 AGV 应使用的预定义动作，如果 AGV 的功能与动作描述相匹配。如果有一种合理的使用定义参数的方式，则应使用它们。如果需要成功执行动作，可以定义附加参数。

如果无法将某些动作映射到下一节中的某个动作，AGV 制造商可以定义供主控使用的附加操作。

6.8.1 预定义动作的定义、参数、效果和范围

通用						范围		
Action 动作	counter action 计数器动作	description 描述	Idempotent 幂等	Parameters 参数	linked state 连接状态	Instant 立刻	node	edge
startPause	stopPause	需要链接状态，因为可以使用硬件开关暂停许多 AGV。不再需要 AGV 行驶运动到达下一个节点。 动作可以继续。 订单可以恢复。	yes	-	paused	yes	no	no
stopPause	startPause	运动和其他所有动作将恢复（如果有）。 需要一个关联状态，因为许多 AGV 可以通过硬件开关暂停。stopPause 也可以重新启动那些使用触发 startPause 的硬件按钮停止的车辆（如果配置了）。	yes	-	paused	yes	no	no
startCharging	stopCharging	启动充电过程。 充电可以在充电站（车辆停放）或充电通道（行驶中）进行。过充保护是车辆的责任。	yes	-	.batteryState.charging	yes	yes	no
stopCharging	startCharging	停止充电过程以发送新订单。 充电过程也可能被车辆/充电站中断，例如当电池充满时。只有当 AGV 准备好接收订单时，电池状态才允许为"false"。	yes	-	.batteryState.charging	yes	yes	no

general						scope		
action	counter action	description	idempotent	parameters	linked state	instant	node	edge
initPosition	-	重置（覆盖）AGV 的位姿，使用给定的参数。	yes	x (float64) y (float64) theta (float64) mapId(string) lastNodeld(string)	.agvPosition.x .agvPosition.y .agvPosition.theta .agvPosition.mapId .lastNodeld .maps	yes	yes (Elevator)	no
enableMap	-	启用先前下载的地图，以便在订单中使用，而无需初始化新位置。	yes	mapId (string) mapVersion (string)	.maps	yes	yes	no
downloadMap	-	触发新地图的下载。下载期间激活。错误报告在车辆状态中。验证下载成功后完成，准备地图使用并将地图设置为状态。	yes	mapId (string) mapVersion (string) mapDownloadLink (string) mapHash (string, optional)	.maps	yes	no	no
deleteMap	-	触发从车辆内存中删除地图	Yes	mapId (string) mapVersion (string)	maps	yes	no	no
stateRequest	-	请求 AGV 发送新的状态报告。	yes	-	-	yes	no	no
logReport	-	请求 AGV 生成并存储日志报告。	yes	reason (string)	-	yes	no	no
pick	drop (if automated)	请求 AGV 取货。具有多个装卸设备的 AGV 可以并行处理多个取货操作。在这种情况下，需要提供参数 lhd（例如，LHD1）。参数 stationType 详细说明取货操作的处理方式（例如，楼层位置、货架位置、被动输送带、主动输送带等）。负载类型说明负载单元，可用于切换现场（例如，EPAL、INDU 等）。	no	lhd (string, optional) stationType (string) stationName(string, optional) loadType (string) loadId(string, optional) height (float64) (optional) defines bottom of the load related to the floor depth (float64) (optional) for	.load	no	yes	yes

general						scope		
action	counter action	description	idempotent	parameters	linked state	instant	node	edge
		为准备装卸设备（例如，基于高度参数的预提升操作），该操作可以提前 horizon 内宣布。但是，预提升操作等不会在 AGV 状态中报告为‘运行’状态，因为相关节点尚未释放。如果车辆在边缘，它可以使用其传感设备来检测拾取节点的位置。		forklifts side(string) (optional) e.g., conveyor				
drop	pick (if automated)	请求 AGV 放下一个负载。查看 pick 获取更多详细信息。	no	lhd (string, optional) stationType (string, optional) stationName (string, optional) loadType (string, optional) loadId(string, optional) height (float64, optional) depth (float64, optional)load	no	yes	yes
detectObject	-	AGV 检测物体（例如，货物、充电站、空闲停车位）。	yes	objectType(string, optional)	-	no	yes	yes
finePositioning	-	在一个节点上，AGV 将精确地定位到目标位置。AGV 允许偏离其节点位置。在一个边上，AGV 在穿越边时，例如，会与静止设备对齐。InstantAction：AGV 开始精确地定位到目标位置。	yes	stationType(string, optional) stationName(string, optional)	-	no	yes	yes

general						scope		
action	counter action	description	idempotent	parameters	linked state	instant	node	edge
waitForTrigger	-	AGV 需要等待 AGV 上的触发信号 (例如, 按钮按下、手动装载)。主控负责处理超时, 并在必要时取消订单。	yes	triggerType(string)	-	no	yes	no
cancelOrder	-	AGV 尽快停止。这可以是立即或在下一个节点。然后订单被删除。所有操作都被取消。	yes	-	-	yes	no	no
factsheetRequest	-	请求 AGV 发送 factsheet	yes	-	-	yes	no	no

6.8.2 预定义的行动定义，对其状态的描述

	action states				
action	'INITIALIZING'	'RUNNING'	'PAUSED'	'FINISHED'	'FAILED'
startPause	-	模式激活正在准备中。 如果 AGV 支持即时转换，则可以省略此状态。	-	车辆静止 所有动作将被暂停 暂停模式被激活 AGV 上报.paused: "true".	由于某种原因（例如，被硬件开关覆盖），暂停模式无法激活。
stopPause	-	正在准备退出该模式。如果 AGV 支持即时转换，则可以省略此状态。	-	暂停模式已停用。 所有的暂停动作将被恢复 AGV 上报 .paused: "false".	由于某种原因（例如，被硬件开关覆盖），暂停模式无法被禁用。
startCharging	-	充电过程的激活正在进行中。	-	充电过程已启动。	无法启动充电过程

	action states					
action	'INITIALIZING'	'RUNNING'	'PAUSED'	'FINISHED'	'FAILED'	
		(正在与充电器通信)。如果 AGV 支持即时转换，则可以省略此状态。		AGV 上报 .batteryState.charging: "true".	某种原因 (例如，未与充电器对齐)。充电问题应与错误对应。	
stopCharging	-	充电过程正在进行中 (正在与充电器通信)。如果 AGV 支持即时转换，可以省略此状态。	-	充电过程已停止。 AGV 上报 .batteryState.charging: "false"	由于某种原因 (例如，未与充电器对齐)，充电过程无法停止。充电问题应与错误对应。	
initPosition	-	新姿态的初始化正在进行中 (进行置信度检查等)。如果 AGV 支持即时转换，可以省略此状态。	-	姿态重置. AGV 上报 .agvPosition.x = x, .agvPosition.y = y, .agvPosition.theta = theta .agvPosition.mapId = mapId .agvPosition.lastNodeId = lastNodeId	姿态无效或无法重置。 一般的定位问题应该对应一个错误。	
downloadMap	连接到地图下载服务器	AGV 正在下载地图，直到下载完成。	-	AGV 通过设置 mapId/mapVersion 和相应的 mapStatus 为 'DISABLED' 来更新其状态。	下载失败，更新到车辆状态 (例如，连接丢失，地图服务器不可达，mapId/mapVersion 不存在。)	

	action states				
action	'INITIALIZING'	'RUNNING'	'PAUSED'	'FINISHED'	'FAILED'
enableMap	-	AGV 启用具有请求的 mapId 和 mapVersion 的地图，同时禁用具有相同 mapId 的其他版本。	-	AGV 将请求的地图的对应 mapStatus 更新为"ENABLED"，并将其他具有相同 mapId 的版本更新为"DISABLED"。	请求的 mapId/mapVersion 组合不存在
deleteMap	-	AGV 从其内部存储中删除了请求的 mapId 和 mapVersion。	-	AGV 从其状态中移除了 mapId/mapVersion。	不能删除地图，如果地图当前正在使用中。所请求的 mapId/mapVersion 组合在之前已经被删除。
stateRequest	-	-	-	状态已被传达	-
logReport	-	报告正在生成 如果 AGV 支持即时生成，则可以省略此状态。	-	报告已存储。日志名称将在状态中报告。	报告无法存储（例如，没有空间）。
pick	初始化取货过程，例如，未完成的举升操作。	取货过程正在运行（AGV 正在进入站点，装卸设备忙碌，与站点通信正在进行等）。	取货过程正在暂停，例如，如果违反了安全字段。删除违规后，挑选过程继续。	取货完成 负载已进入 AGV，AGV 报告新的负载状态。	取货失败，例如，站点意外为空。失败的取货操作应与错误对应。

	action states				
action	'INITIALIZING'	'RUNNING'	'PAUSED'	'FINISHED'	'FAILED'
drop	放货过程的初始化，例如，未完成的下降操作。	放货过程正在运行 (AGV 正在进入站点，负载处理设备繁忙，与站点通信正在进行中，等等)。	放货过程正在暂停，例如：如果违反了安全字段。删除违规后，放弃过程继续。	放货完成。负载已经离开 AGV，AGV 上报新的负载状态	放货失败，例如，站点意外占用。失败的放下操作应与错误对应。
detectObject	-	目标检测正在执行	-	对象已经被识别	AGV 不能够识别对象
finePositioning	-	AGV 精确地定位在目标位置上。	精细定位过程正在暂停，例如，如果违反了安全字段。消除违规行为后，继续进行精细定位。	相对于站点的目标位置已达到。	参考站位置无法到达。
waitForTrigger	-	AGV 正在等待触发	-	触发器已经被触发	如果订单已取消，waitForTrigger 将失败。
cancelOrder	-	AGV 停止或行驶，直到到达下一个节点。	-	AGV 停止并已取消订单。	-
factsheetRequest	-	-	-	已传达 factsheet	-

6.9 主题: "instantActions" (从主控到控制到 AGV)

在某些情况下，需要向 AGV 发送需要立即执行的操作。这是通过向 instantActions 主题发布 instantAction 消息来实现的。instantActions 不得与 AGV 当前订单的内容冲突（例如，发送降低叉车操作，而订单要求提升叉车）。

以下是一些可能需要即时操作的情况：

- 暂停 AGV，而不更改当前订单中的任何内容；
- 暂停后恢复订单；
- 激活信号（光学、音频等）。

对于更多信息，请参阅第 7 节最佳实践。

Object structure	Data type	Description
headerId	uint32	Header ID of the message. The header ID is defined per topic and incremented by 1 with each sent (but not necessarily received) message.
timestamp	string	Timestamp (ISO 8601, UTC); YYYY-MM-DDTHH:mm:ss.ffZ (e.g., "2017-04-15T11:40:03.12Z")
version	string	Version of the protocol [Major].[Minor].[Patch] (e.g., 1.3.2).
manufacturer	string	Manufacturer of the AGV.
serialNumber	string	Serial number of the AGV.
actions [action]	array	需要立即执行且不属于常规订单的动作数组。

当 AGV 接收到 instantAction 时，会在 AGV 状态的 actionStates 数组中添加适当的 actionStatus。根据动作的进度更新 actionStatus。有关 actionStatus 的不同转换，请参见图 16。

6.10 主题: "state" (从 AGV 到主控)

AGV 状态将通过一个主题进行传输。与使用单独消息（例如，用于订单，电池状态和错误）相比，使用一个主题将减少代理和主控处理消息的工作量，同时保持 AGV 状态信息的同步。

AGV 状态消息将在相关事件发生时或最迟每 30 秒通过 MQTT 代理发布到主控。

触发状态消息传输的事件包括：

- 接收订单
- 接收订单更新
- 负载状态发生变化
- 错误或告警
- 驶过节点
- 切换操作模式

- Driving 字段的变化
- nodeStates、edgeStates、actionStates 或 maps 的变化

应努力控制通信量。如果两个事件相互关联（例如，接收新订单通常会强制更新节点和边状态；同样，经过一个节点也会触发状态更新），则触发一次状态更新比多次更新更合理。

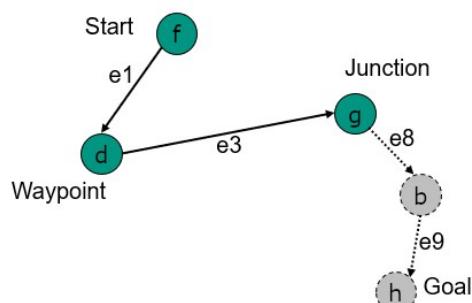
6.10.1 概念和逻辑

订单进度由 nodeStates 和 edgeStates 跟踪。此外，如果 AGV 能够推导出其当前位置，它可以通过位置字段发布其位置。

如果 AGV 自行规划路径，它应当通过状态消息中的 trajectory 对象以 NURBS 形式通信其计算出的轨迹（包括 base 和 horizon），除非主控无法使用该字段，并且在集成过程中已达成协议，该字段不应发送。在主控释放节点后，AGV 不允许更改其轨迹。

nodeStates 和 edgeStates 包括 AGV 仍需遍历的所有节点/边。

Graph transmitted to AGV in topic order



Feedback from AGV in topic state

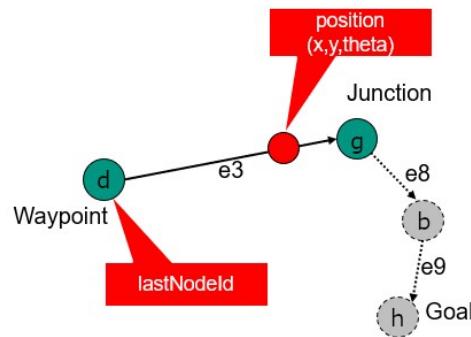


图 14 状态主题提供的订单信息。仅传输最后一个节点的 ID 以及剩余的节点和边

6.10.2 节点遍历及进入/离开边，触发动作

AGV 自行决定何时将节点视为已通过。通常，AGV 的控制点应在节点的 allowedDeviationXY 范围内，其方向应在 allowedDeviationTheta 范围内。如果后续边的 edge 属性 corridor 被设置，则应额外满足这些边界条件。

AGV 通过从 nodeStates 数组中移除其 nodeState 并设置 lastNodeId、lastNodeSeqId 为已通过节点的值来报告节点的通过。

一旦 AGV 报告该节点已遍历，AGV 应触发与该节点相关的操作（如果有）。

节点的通过也标志着到达该节点的边的离开。该边应从 edgeStates 中移除，并且在该边上激活的操作应完成。

节点的遍历也标志着 AGV 进入下一条边（如果存在）的时刻。现在应触发该边的操作。此规则的例外情况是，如果 AGV 需要在边上暂停（由于软阻塞或硬阻塞或其他原因）——那么 AGV 在重新开始移动后进入该边。

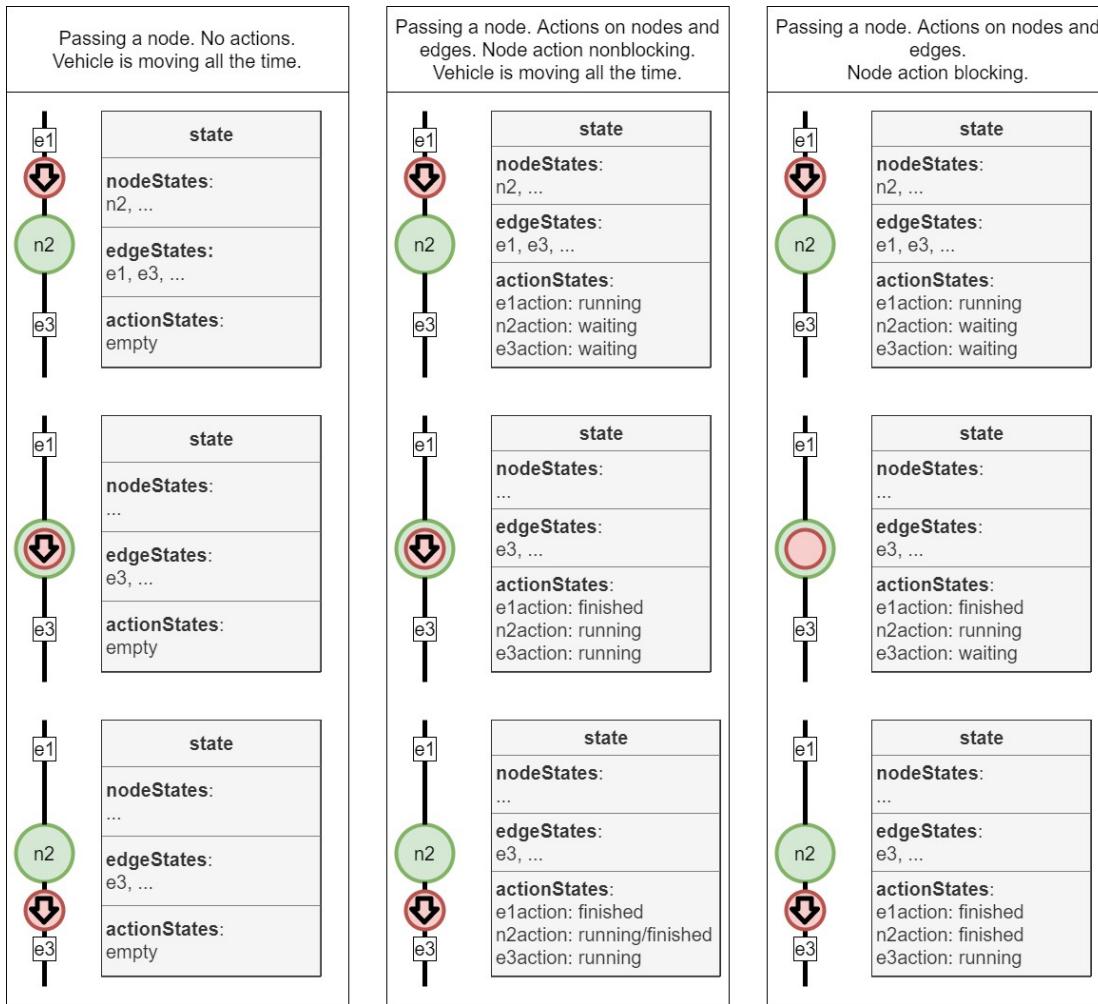


图 15 订单处理期间 nodeStates, edgeStates, actionStates 的示意图

6.10.3 基础请求

如果 AGV 检测到其电池电量不足，可以设置 newBaseRequest 标志为"true"以防止不必要的制动。

6.10.4 信息

AGV 可以通过信息数组向主控提交任意额外的信息。AGV 通过信息消息报告信息的时长由 AGV 决定。

主控不应使用信息消息进行逻辑处理，它仅用于可视化和调试目的。

6.10.5 错误

AGV 通过错误数组报告错误。错误分为两个级别：'WARNING'和'FATAL'。'WARNING'是一种可自行解决的错误，例如字段违规。'FATAL'错误需要人工干预。错误可以通过错误引用数组传递引用，以帮助找到错误的根本原因。

6.10.6 状态消息的实现

Object structure	Unit	Data type	Description
headerId		uint32	Header ID of the message. The headerId is defined per topic and incremented by 1 with each sent (but not necessarily received) message.
timestamp		string	Timestamp (ISO 8601, UTC); YYYYMM-DDTHH:mm:ss.ffZ (e.g., "201704-15T11:40:03.12Z").
version		string	Version of the protocol [Major].[Minor].[Patch] (e.g., 1.3.2).
manufacturer		string	Manufacturer of the AGV.
serialNumber		string	Serial number of the AGV.
maps[map]		array	Array of map objects that are currently stored on the vehicle.
orderId		string	当前订单或之前完成的订单的唯一订单标识。orderId 保持不变，直到收到新订单。如果没有之前的 orderId，则为空字符串("")。
orderUpdateId		uint32	订单更新识别，以识别 AGV 已接受订单更新。"0" 则之前没有可用的 orderUpdateId。
zoneSetId		string	区域集的唯一 ID，AGV 目前用于路径规划。应该和订单中使用的一样。 可选：如果 AGV 不使用区域，则可以省略此字段。
lastNodeId		string	最后到达节点的节点 ID，或者，如果 AGV 当前在节点上，则当前节点（例如“node7”）。如果没有可用的 lastNodeId，则为空字符串（“ ”）。
lastNodeSequenceId		uint32	最后到达的节点的序列 ID，或者，如果 AGV 当前在节点上，则当前节点的序列 ID。如果没有可用的 lastNodeSequenceId，则为“0”。
nodeStates [nodeState]		array	需要遍历以完成订单的 nodeState 对象数组（如果闲置，则为空数组）
edgeStates [edgeState]		array	需要遍历以完成订单的边状态对象数组（如果闲置，则为空数组）

Object structure	Unit	Data type	Description
agvPosition		JSON object	地图中 AGV 的当前位置。 可选: 仅当 AGV 没有定位能力时可以省略，例如线引导 AGV。
velocity		JSON object	AGV 在车辆坐标系中的速度。
loads [load]		array	当前处理的负载数组。 可选:如果 AGV 无法确定负载状态，则应完全省略此字段，不报告为空数组。如果 AGV 能够确定负载状态，但数组为空，则认为 AGV 处于无负载状态。
driving		boolean	“true”: 表示 AGV 正在驱动和/或旋转。此处不包括 AGV 的其他动作（例如提升动作）。“false”: 表示 AGV 既不行驶也不旋转。
paused		boolean	“true” : AGV 目前处于暂停状态，要么是因为按下 AGV 上的物理按钮，要么是因为 instantAction。AGV 可以恢复订单。 "false": AGV 当前不处于暂停状态
newBaseRequest		boolean	“true”: AGV 几乎在 base 的末端，如果没有传输新的 base，将降低速度。 触发主控制以发送新 base。 “False”: 无需更新 base。
distanceSinceLastNode	meter	float64	线路引导车辆用来表示它经过 “lastNodeId”的行驶距离。距离以米为单位。

Object structure	Unit	Data type	Description
actionStates [actionState]		array	<p>包含从当前订单和自上次订单以来所有接收到的 instantActions 的所有动作的数组。</p> <p>actionStates 会一直保持，直到接收到新订单。在接收到新订单时，除了正在运行的 instantActions 之外，其他 actionStates 将被移除。这可能包括来自先前节点的、仍在进行中的动作。</p> <p>当一项操作完成时，会发布一个更新状态的消息，其中将 actionStatus 设置为'FINISHED'，如果适用的话，还会包含相应的 resultDescription.</p>
batteryState		JSON object	包含电池相关的所有信息
operatingMode		string	<p>枚举值{'AUTOMATIC', 'SEMAUTOMATIC', 'MANUAL', 'SERVICE', 'TEACHIN'}</p> <p>更多信息，请参见第 6.10.6 节状态消息的实现中的表 1。</p>
errors [error]		array	<p>错误的对象数组</p> <p>所有 AGV 的激活错误都应在此数组中。空数组表示 AGV 没有激活错误。</p>
information [info]		array	<p>信息的对象数组</p> <p>空数组表示 AGV 没有信息。这仅应用于可视化或调试——它不应用于主控逻辑。</p>
safetyState		JSON object	包含所有安全相关的信息

Object structure	Unit	Data type	Description
map{		JSON object	
mapId		string	描述车辆工作区定义区域的地图 ID。
mapVersion		string	地图版本
mapDescription		string	地图的附加信息

Object structure	Unit	Data type	Description
mapStatus }		string	枚举值{'ENABLED', 'DISABLED'} 'ENABLED': 表示此地图当前在 AGV 上处于活动/使用状态。最多只有一个具有相同 mapId 的地图可以将其状态设置为'ENABLED'。 'DISABLED': 表示此地图版本当前在 AGV 上未启用，因此可以根据请求启用或删除。

Object structure	Unit	Data type	Description
nodeState {		JSON object	
nodeId		string	唯一的节点定义
sequenceId		uint32	识别具有相同节点 ID 的多个节点的序列 ID。
nodeDescription		string	节点的附加信息
released		boolean	"true"表示节点是 base 的一部分。 "false"表示节点是 horizon 的一部分。
nodePosition }		JSON object	节点位置 该对象定义在 6.6 主题: "order" (从主控到 AGV) 可选: 主控具有此信息。可以额外发送，例如用于调试目的。

Object structure	Unit	Data type	Description
edgeState {		JSON object	
edgeId		string	边的唯一标识
sequenceId		uint32	序列 ID 用于区分具有相同 edgeId 的多个边。
edgeDescription		string	边的附加信息
released		boolean	"true" 表示边是 base 的一部分。 "false"表示边是 horizon 的一部分。
trajectory }		JSON object	轨迹将作为 NURBS 进行通信，并在 6.6.6 订单消息的实现 轨迹段从车辆进入边的点开始，到车辆报告已通过终点节点时结束。

Object structure	Unit	Data type	Description
agvPosition {		JSON object	以世界坐标定义地图上的位置。每层楼都有自己的地图。
positionInitialized		boolean	“true”: 位置已初始化。 “false”: 位置未初始化。
localizationScore		float64	范围: [0.0 ... 1.0] 描述定位质量, 因此, 例如, SLAM AGV 可以使用它来描述当前位置信息的准确性。0.0: 位置未知 1.0: 位置已知 对于无法估计其定位分数的车辆是可选的。仅用于日志记录和可视化。
deviationRange	m	float64	位置偏差范围的值, 以米为单位。 对于无法估计其偏差的车辆 (例如基于网格的定位), 此属性可选。仅用于记录和可视化目的。
x	m	float64	参考地图坐标系的地图上的 X 位置。 精确度取决于具体的实现。
y	m	float64	参考地图坐标系的地图上的 Y 位置。 精确度取决于具体的实现。
theta		float64	范围: [-Pi ... Pi] AGV 的朝向
mapId		string	引用位置的地图的唯一识别。每张地图都有相同的坐标原点。 当 AGV 使用电梯从起始楼层前往目标楼层时, 它会离开起始楼层的地图, 并在目标楼层的地图上相应的电梯节点处重生。
mapDescription }		string	地图的附加信息

Object structure	Unit	Data type	Description
velocity {		JSON object	
vx	m/s	float64	AGV 在 X 方向的速度

Object structure	Unit	Data type	Description
<i>vy</i>	m/s	float64	AGV 在 Y 方向的速度
<i>omega</i> }	Rad/s	float64	AGV 绕 Z 轴的转向速度。

Object structure	Unit	Data type	Description
load {		JSON object	
<i>loadId</i>		string	负载的唯一识别 (例如, 条形码或 RFID)。 空字段, 如果 AGV 可以识别负载, 但尚未识别负载。 如果 AGV 无法识别负载, 则可选。
<i>loadType</i>		string	负载类型
<i>loadPosition</i>		string	指示使用 AGV 的哪个负载处理/承载单元, 例如, 如果 AGV 有多个点/位置来承载负载。例如: “前”、“后”、“位置 C1”等。对于只有一个负载位置的车辆可选
boundingBoxReference		JSON object	边界框位置的参考点。参考点始终是边界框底面的中心 (高度=0), 并用 AGV 坐标系的坐标来描述。
loadDimensions		JSON object	负载边界框的尺寸, 单位为米
<i>weight</i> }	kg	float64	范围: [0.0 ... float64.max] 负载的绝对重量, 单位为千克。

Object structure	Unit	Data type	Description
boundingBoxReference {		JSON object	参考点, 用于确定边界框的位置。 参考点始终是边界框底部中心 (高度=0), 并以 AGV 坐标系中的坐标描述。
x		float64	X-coordinate of the point of reference.
y		float64	Y-coordinate of the point of reference.

Object structure	Unit	Data type	Description
z		float 64	Z-coordinate of the point of reference.
theta }		float64	负载边界框的方向。 对牵引车、火车等非常重要。

Object structure	Unit	Data type	Description
loadDimensions {		JSON object	负载边界框的尺寸，单位为米。
length	m	float64	负载边界框的绝对长度。
width	m	float64	负载边界框的绝对宽度。
height }	m	float64	负载边界框的绝对高度。 可选： 仅当已知时设置值。

Object structure	Unit	Data type	Description
actionState {		JSON object	
actionId		string	动作的唯一标识
actionType		string	动作的类型 可选：仅用于信息或可视化目的。MC 知晓在订单中分派的动作类型。
actionDescription		string	当前动作的附加信息
actionStatus		string	枚举值{"WAITING", "INITIALIZING", "RUNNING", "PAUSED", "FINISHED", "FAILED"} 参见 6.11 动作状态
resultDescription }		string	结果描述，例如 RFID 读取的结果。 错误将被传输在错误中。

Object structure	Unit	Data type	Description
batteryState {		JSON object	
batteryCharge	%	float64	充电状态：如果 AGV 仅提供良好或不良电池电量的值，这些将被表示为 20% (不良) 和 80% (良好)。

Object structure	Unit	Data type	Description
<i>batteryVoltage</i>	V	float64	电池电压
<i>batteryHealth</i>	%	int8	范围: [0 ... 100] 描述了电池的监控状态
<i>charging</i>		boolean	“true”: 充电中。 “false”: AGV 当前没有在充电
<i>reach</i> }	m	uint32	范围: [0 ... uint32.max] 当前电量状态下的预计到达的范围。

Object structure	Unit	Data type	Description
error {		JSON object	
errorType		string	错误的类型/名称
errorReferences [errorReference]		array	引用数组 (例如, <code>nodeId</code> 、 <code>edgeId</code> 、 <code>orderId</code> 、 <code>actionId</code> 等) , 以提供与错误相关的更多信息。 有关更多信息, 请参阅 7 个最佳做法。
errorDescription		string	提供错误详细信息和可能原因的冗长描述。
errorHint		string	关于如何处理或解决所报告错误的提示。
errorLevel }		string	枚举值{"WARNING", 'FATAL'} 'WARNING': AGV 准备开始 (例如, 维护周期到期警告)。'FATAL': AGV 不在运行状态, 需要用户干预 (例如, 激光扫描仪被污染)。

Object structure	Unit	Data type	Description
errorReference {		JSON object	
referenceKey		string	指定使用的引用类型 (例如, “ <code>nodeId</code> ”、“ <code>edgeId</code> ”、“ <code>orderId</code> ”、“ <code>actionId</code> ”等) 。
referenceValue }		string	属于参考键的值。例如, 出现错误的节点的 ID。

Object structure	Unit	Data type	Description
info {		JSON object	
infoType		string	信息的类型/名称
<i>infoReferences [infoReference]</i>		array	信息参考数组
<i>infoDescription</i>		string	信息描述
<i>infoLevel }</i>		string	枚举值{'DEBUG', 'INFO'} 'DEBUG': 用于调试 'INFO': 用于可视化

Object structure	Unit	Data type	Description
infoReference {		JSON object	
referenceKey		string	消息引用引用类型 (例如, headerId、orderId、actionId 等)。
<i>referenceValue }</i>		string	该值属于参考键。

Object structure	Unit	Data type	Description
safetyState {		JSON object	
eStop		string	枚举值 {'AUTOACK', 'MANUAL', 'REMOTE', 'NONE'} eStop 的确认类型: 'AUTOACK': 自动确认的紧急停止按 钮已激活, 例如由缓冲器或防护场触 发。 'MANUAL': 紧急停止按钮需要在车辆 处手动确认。 'REMOTE': 设施紧急停止按钮需要远 程确认。 'NONE': 未激活紧急停止。
<i>fieldViolation }</i>		boolean	字段违规保护。 "true": 字段违规 "false": 字段没有违规

操作模式描述

下列描述列出了主题"state"的 operatingMode。

Identifier	Description
AUTOMATIC	AGV 完全受主控控制。 AGV 根据主控的指令进行驾驶和执行动作。
SEMITAUTOMATIC	AGV 半自动受主控控制。 AGV 根据主控的指令行驶并执行动作。 行驶速度由 HMI 控制（速度不能超过自动模式的速度）。 转向由自动控制（可能存在非安全 HMI）。
MANUAL	主控不控制 AGV。 监控器不向 AGV 发送行驶指令或动作。HMI 可用于控制 AGV 的方向、速度和处理设备。 AGV 的位置发送到主控。 当 AGV 进入或离开此模式时，它会立即清除所有指令（需要安全 HMI）。
SERVICE	主控不控制 AGV。 主控不向 AGV 发送行驶指令或动作。授权人员可以重新配置 AGV。
TEACHIN	主控不控制 AGV。 监督者不向 AGV 发送行驶指令或动作。 AGV 正在被教学，例如，地图绘制由主控完成。

表 1 操作模式及其含义

6.11 动作状态

当 AGV 接收到一个动作（附加到节点或边缘或通过 instantAction）时，它应在其 actionStates 数组中用 actionState 表示此操作。

actionStates 字段中的 actionStatus 描述了动作在其生命周期中的哪个阶段。

表 2 描述了枚举 actionStatus 可以持有的值。

actionStatus	Description
'WAITING'	动作被 AGV 接收，触发该动作的节点尚未到达或激活该动作的边尚未进入。
'INITIALIZING'	动作被触发，预备措施已启动。
'RUNNING'	动作已经在执行
'PAUSED'	动作因 pause instantAction 或外部触发而暂停。（AGV 上的暂停按钮）
'FINISHED'	动作已完成。通过 resultDescription 报告结果。
'FAILED'	Action could not be finished for whatever reason.

表 2 actionStatus 字段的接受值

状态转换图如图 16 所示。

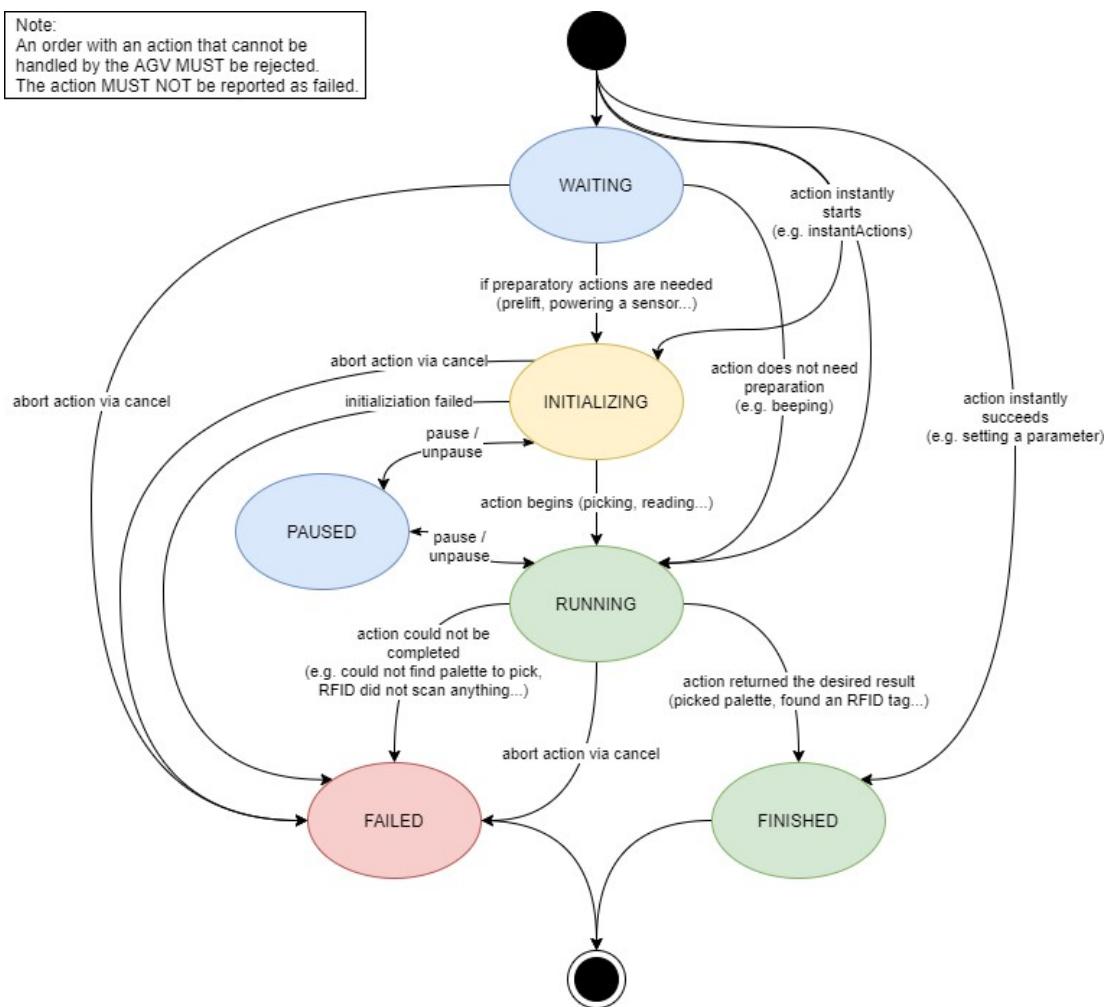


图 16 动作状态所有可能的转换

6.12 动作阻塞类型和顺序

订单中多个动作的顺序定义了这些动作要执行的序列。动作的并行执行由各自的 **blockingType** 控制。

动作可以有三种不同的阻塞类型，如表 3 所述。

actionStatus	Description
NONE	动作可与其它动作并行执行，且车辆可行驶。
SOFT	动作可与其它动作并行执行。车辆不应行驶。
HARD	动作不应与其它动作并行执行。车辆不应行驶。

表 3 动作阻塞类型

如果同一节点上有多个不同阻塞类型的动作，图 17 描述了 AGV 应如何处理这些动作。

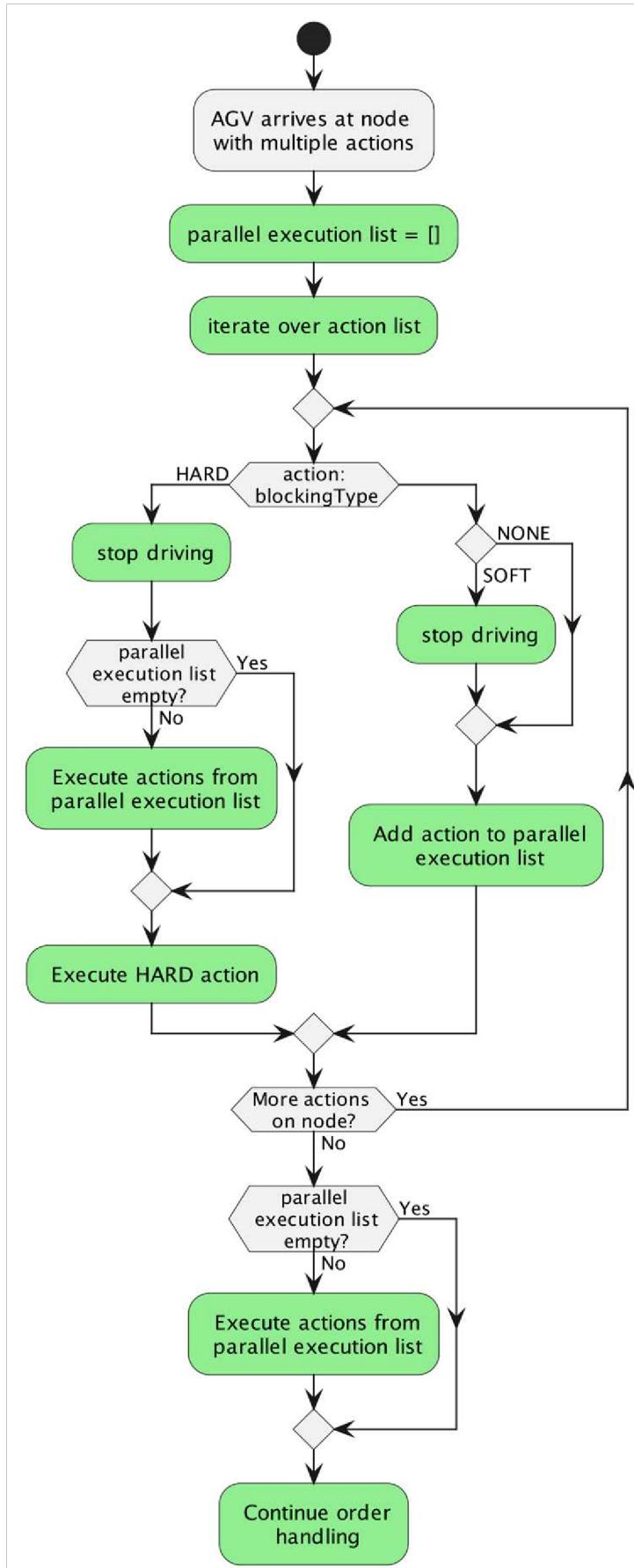


图17 处理多个动作

6.13 主题 "visualization"

为了近乎实时地更新位置，AGV 可以在可视化主题上广播其位置和速度。

位置对象的结构与状态中的位置和速度对象相同。有关更多信息，请参阅第 6.10.6 状态消息的实现。此主题的更新速率由集成商定义。

6.14 主题 "connection"

在 AGV 客户端连接到代理时，可以设置一个"last will"主题和消息，当 AGV 客户端与代理断开连接时，代理会发布该主题和消息。因此，主控可以通过订阅所有 AGV 的连接主题来检测断开连接事件。断开连接通过代理和客户端之间交换的心跳来检测。在大多数代理中，该间隔是可配置的，应设置为大约 15 秒。连接主题的服务质量级别应为 1 - 至少一次。

建议 last will 主题结构是：

`uagv/v2/manufacturer/SN/connection`

last will 消息被定义为包含以下字段的 JSON 封装消息：

Identifier	Data type	Description
headerId	uint32	Header ID of the message. The headerId is defined per topic and incremented by 1 with each sent (but not necessarily received) message.
timestamp	string	Timestamp (ISO8601, UTC); YYYY-MM-DDTHH:mm:ss.ffZ(e.g., "2017-04-15T11:40:03.12Z").
version	string	Version of the protocol [Major].[Minor].[Patch] (e.g., 1.3.2).
manufacturer	string	Manufacturer of the AGV.
serialNumber	string	Serial number of the AGV.
connectionState	string	枚举值 {'ONLINE', 'OFFLINE', 'CONNECTIONBROKEN'} 'ONLINE': AGV 与代理之间的连接处于活动状态。 'OFFLINE': AGV 与代理之间的连接以协调方式离线。 'CONNECTIONBROKEN': AGV 与代理之间的连接意外中断。

当使用 MQTT 断开连接命令以优雅方式结束连接时，最后 last will 消息不会发送。只有当连接意外中断时，代理才会发送最后 last will 消息。

注意：由于 MQTT last will 功能的特性，最后 last will 消息是在 AGV 与 MQTT 代理的连接阶段定义的。因此，时间戳和 headerId 字段将始终过时。

AGV 希望优雅的断开连接：

1. AGV 发送 "uagv/v2/manufacturer/SN/connection" 将 connectionState 设置成 OFFLINE.

2. 使用断开命令断开 MQTT 连接。

AGV 上线:

1. 当 MQTT 连接创建时, 设置 last will 消息到 "uagv/v2/manufacturer/SN/connection" 并将字段 connectionState 设置成 CONNECTIONBROKEN
2. 发送主题 "uagv/v2/manufacturer/SN/connection" 将 connectionState 设置 ONLINE.

与此主题相关的所有消息都应带有保留标志。

当 AGV 与代理之间的连接意外中断时, 代理将发送 last will 主题:

"uagv/v2/manufacturer/SN/connection", 并将字段 connectionState 设置为 CONNECTIONBROKEN。

6.15 主题 "factsheet"

产品简介提供了关于特定 AGV 系列的基本信息。这些信息允许比较不同的 AGV 类型, 并可用于 AGV 系统的规划、尺寸设计和仿真。产品简介还包括有关 AGV 通信接口的信息, 这些接口是 AGV 系列集成到符合 VDA-5050 标准的中央控制系统所必需的。

AGV 产品简介中某些字段的值只能在系统集成过程中指定, 例如特定项目的负载和站点类型的分配, 以及该 AGV 支持的站点和负载类型列表。

该 factsheet 旨在既作为人类可读文档, 也用于机器处理, 例如由主控应用程序导入, 因此指定为 JSON 文档。

MC 可以通过发送即时动作向 AGV 请求 factsheet:

factsheetRequest

与此主题相关的所有消息都应带有保留标志。

6.15.1 Factsheet JSON 结构

该 factsheet 由以下表格中列出的 JSON 对象组成。

Field	data type	description
headerId	uint32	Header ID of the message. The headerId is defined per topic and incremented by 1 with each sent (but not necessarily received) message.
timestamp	string	Timestamp (ISO8601, UTC); YYYY-MMDDTHH:mm:ss.ffZ (e.g., "2017-0415T11:40:03.12Z").
version	string	Version of the protocol [Major].[Minor].[Patch] (e.g., 1.3.2).
manufacturer	string	Manufacturer of the AGV.
serialNumber	string	Serial number of the AGV.

Field	data type	description
typeSpecification	JSON object	这些参数通常指定 AGV 的类别和功能。
physicalParameters	JSON object	这些参数指定 AGV 的基本物理属性。
protocolLimits	JSON object	MQTT 通信中标识符、数组、字符串和类似项目的长度限制。
protocolFeatures	JSON object	VDA5050 协议支持的功能。
agvGeometry	JSON object	AGV 几何的详细定义。
loadSpecification	JSON object	负载能力的抽象规范。
vehicleConfig	JSON object	车辆上当前软件和硬件版本的摘要以及可选的网络信息。

typeSpecification

该 JSON 对象描述了 AGV 类型的通用属性。

Field	data type	description
seriesName	string	制造商指定的通用系列名称。
seriesDescription	string	AGV 类型系列的易读描述。
agvKinematic	string	AGV 运动学类型的简化描述。 [DIFF, OMNI, THREEWHEEL] DIFF: 差速驱动 OMNI: 全向车辆 THREEWHEEL: 三轮驱动车辆或具有类似运动学的车辆
agvClass	string	AGV 类别的简短描述。 [FORKLIFT, CONVEYOR, TUGGER, CARRIER] FORKLIFT: forklift. CONVEYOR: 带有传送带的 AGV。 TUGGER: 牵引车。 CARRIER: 带或不带提升单元的货物载具。
maxLoadMass	float64	[kg], 最大可承载质量。
localizationTypes	array of string	定位类型的简化描述。示例值：NATURAL: 自然地标; REFLECTOR: 激光反射器; RFID: RFID 标签; DMC: 数据矩阵码; SPOT: 磁点; GRID: 磁栅.
navigationTypes	array of string	按优先级排序的 AGV 支持的路由规划类型数组。 示例值：PHYSICAL_LINE_GUIDED: 无路径规划, AGV 沿物理安装的路径行驶。 VIRTUAL_LINE_GUIDED: AGV 沿固定 (虚拟) 路径行驶。AUTONOMOUS: AGV 自主规划路径。

physicalParameters

该 JSON 对象描述了 AGV 的物理属性。

Field	data type	description
speedMin	float64	[m/s] AGV 的最小控制连续速度。
speedMax	float64	[m/s] AGV 的最大速度。
angularSpeedMin	float64	[Rad/s] AGV 的最小控制连续旋转速度。
angularSpeedMax	float64	[Rad/s] AGV 的最大旋转速度。
accelerationMax	float64	[m/s ²]最大负载时的最大加速度。
decelerationMax	float64	[m/s ²]最大负载时的最大减速度。
heightMin	float64	[m] AGV 的最小高度。
heightMax	float64	[m] AGV 的最大高度。
width	float64	[m] AGV 的宽度。
length	float64	[m] AGV 的长度。

protocolLimits

这个 JSON 对象描述了 AGV 的协议限制。如果某个参数未定义或设置为 0，则该参数没有明确的限制。

Field	data type	description
maxStringLens {	JSON object	字符串的最大长度
msgLen	uint32	最大 MQTT 消息长度。
topicSerialLen	uint32	MQTT-topics 中序列号部件的最大长度 影响参数:order.serialNumberinstantActions.serialNumberstate.SerialNumbervisualization.serialNumberconnection.serialNumber
topicElemLen	uint32	MQTT 主题中所有其他部分的最大长度。 影响参数: order.timestamp order.version order.manufacturer instantActions.timestamp instantActions.version instantActions.manufacturer state.timestamp state.version state.manufacturer visualization.timestamp visualization.version visualization.manufacturer connection.timestamp connection.version connection.manufacturer

Field	data type	description
<i>idLen</i>	uint32	ID 字符串的最大长度。 影响参数: order.orderId order.zoneSetId node.nodeId nodePosition.mapId action.actionId edge.edgeId edge.startNodeId edge.endNodeId
<i>idNumericalOnly</i>	boolean	如果 “true” ID 字符串只需要包含数值。
<i>enumLen</i>	uint32	枚举值和关键字符串的最大长度。 影响参数: action.actionType action.blockingType edge.direction actionParameter.key state.operatingMode load.loadPosition load.loadType actionState.actionStatus error.errorType error.errorLevel errorReference.referenceKey info.infoType info.infoLevel safetyState.eStop connection.connectionState
<i>loadIdLen</i>	uint32	最大负载 ID 字符串长度。
}		
maxArrayLens {	JSON object	数组最大长度。
<i>order.nodes</i>	uint32	每个订单可由 AGV 处理的节点最大数量。
<i>order.edges</i>	uint32	每个订单可由 AGV 处理的边最大数量。
<i>node.actions</i>	uint32	每个节点可由 AGV 处理的动作最大数量。
<i>edge.actions</i>	uint32	每条边可由 AGV 处理的动作最大数量。
<i>actions.actionsParameters</i>	uint32	每个动作可由 AGV 处理的参数最大数量。
<i>instantActions</i>	uint32	每条消息可由 AGV 处理的即时动作最大数量。
<i>trajectory.knotVector</i>	uint32	每条轨迹可由 AGV 处理的节点最大数量。
<i>trajectory.controlPoints</i>	uint32	每条轨迹可由 AGV 处理的控制点最大数量。
<i>state.nodeStates</i>	uint32	AGV 发送的节点状态的最大数量, AGV 基础上的节点的最大数量。

Field	data type	description
<i>state.edgeStates</i>	uint32	AGV 发送的边状态的最大数量, AGV base 的最大边数量。
<i>state.loads</i>	uint32	自动导引车发送的负载对象的最大数量。
<i>state.actionStates</i>	uint32	自动导引车发送的 actionStates 的最大数量。
<i>state.errors</i>	uint32	最大 AGV 在一个状态消息中发送的错误数量。
<i>state.information</i>	uint32	AGV 在一个状态消息中发送的最大信息数量。
<i>error.errorReferences</i>	uint32	AGV 为每个错误发送的最大错误引用数量。
<i>information.infoReferences</i>	uint32	AGV 为每个信息发送的最大信息参考数量。
}		
timing {	JSON object	定时信息。
<i>minOrderInterval</i>	float32	[s],向 AGV 发送订单消息的最小间隔。
<i>minStateInterval</i>	float32	[s],发送状态消息的最小间隔。
<i>defaultStateInterval</i>	float32	[s],发送状态消息的默认间隔, 如果没有定义, 将使用主文档的默认值。
<i>visualizationInterval</i>	float32	[s],发送关于 visualization 主题消息的默认间隔。
}		

protocolFeatures

该 JSON 对象定义了 AGV 支持的动作和参数。

Field	data type	description
optionalParameters [optionalParameter]	array of JSON object	支持和/或必需的可选参数的数组。此处未列出的可选参数假定 AGV 不支持。
{		
<i>parameter</i>	string	可选参数的全名 例如, "order.nodes.nodePosition.allowedDeviationTheta".
<i>support</i>	enum	对可选参数的支持类型, 以下值是可能的: 'SUPPORTED': 支持指定的可选参数。 'REQUIRED': 适当的 AGV 操作需要可选参数。

Field	data type	description
<i>description</i>	string	自由文本：可选参数的描述， <ul style="list-style-type: none"> - 说明为何此 AGV 类型需要可选参数 direction，以及它可以包含哪些值。 - 参数 nodeMarker 应仅包含无符号整数。 - NURBS 支持仅限于直线和圆弧段。
}		
agvActions [agvAction]	array of JSON object	该 AGV 支持的参数化所有动作数组。这包括 VDA5050 中指定的标准动作和制造商特定的动作。
{		
<i>actionType</i>	string	与 action.actionType 对应的唯一动作类型。
<i>actionDescription</i>	string	自由格式文本：动作描述。
<i>actionScopes</i>	array of enum	使用此动作类型的允许范围数组。 'INSTANT': 可用作 instantAction。 'NODE': 可用于节点 'EDGE': 可用于边 例如: ['INSTANT', 'NODE']
actionParameters [actionParameter]	array of JSON object	一个操作所具有的参数数组。 如果未定义，则该操作没有参数。此处定义的 JSON 对象与第 6.6.6 订单消息的实现所使用的 JSON 对象不同。
{		
<i>key</i>	string	参数的关键字符串。
<i>value DataType</i>	enum	值的数据类型，可能的数据类型是: 'BOOL', 'NUMBER', 'INTEGER', 'FLOAT', 'STRING', 'OBJECT', 'ARRAY'.
<i>description</i>	string	自由文本：参数的描述。
<i>isOptional</i>	boolean	"true": 可选参数
}		
<i>resultDescription</i>	string	自由文本：结果描述。
<i>blockingTypes</i>	array of enum	定义动作的可能阻塞类型数组。枚举 {'NONE', 'SOFT', 'HARD'}
}		

agvGeometry

JSON 对象定义了 AGV 的几何属性，例如轮廓和轮子位置。

Field	data type	description
wheelDefinitions [wheelDefinition]	array of JSON object	轮子数组，包含轮子排列和几何信息。
{		
type	enum	轮子类型，枚举 {'DRIVE', 'CASTER', 'FIXED', 'MECANUM'}。
isActiveDriven	boolean	"true": 车轮被主动驱动。
isActiveSteered	boolean	"true": 车轮被主动转向。
position {	JSON object	
x	float64	[m], x 位置在 AGV 坐标系中
y	float64	[m], y 位置在 AGV 坐标系中
theta	float64	[rad], 轮子在 AGV 坐标系中的方向。对于固定轮子是必要的。
}		
diameter	float64	[m], 轮子的标称直径。
width	float64	[m], 轮胎标称宽度。
centerDisplacement	float64	[m], 车轮中心到旋转点的标称位移（脚轮需要）。如果参数未定义，则假定为 0。
constraints	string	自由格式文本：制造商可用于定义约束。
}		
envelopes2d [envelope2d]	array of JSON object	二维 AGV 包络曲线数组，例如，空载和满载状态下的机械包络，不同速度情况下的安全区域。
{		
set	string	包络曲线集的名称。
polygonPoints [polygonPoint]	array of JSON object	包络曲线作为 x/y 多边形时，假定它是闭合的，且不应自相交。
{		
x	float64	[m], 多边形点的 X 坐标。
y	float64	[m], 多边形点的 Y 坐标。
}		
description	string	自由格式文本：包络曲线集的描述。
}		

Field	data type	description
envelopes3d [envelope3d]	array of JSON object	3D 中 AGV 包络曲线数组。
{		
set	string	包络曲线集的名称。
format	string	数据格式, 例如 DXF。
data	JSON object	3D 包络曲线数据, 格式由'format'指定。
url	string	下载 3D 包络曲线数据的协议和 URL 定义, 例如 ftp://xxx.yyy.com/ac4dgvhoif5tghji 。
description	string	自由格式文本: 包络曲线集的描述
}		

loadSpecification

JSON 对象指定了自动导引车 (AGV) 的负载处理和支持的负载类型。

Field	data type	description
loadPositions	array of string	负载位置/负载处理设备的数组。该数组包含参数"state.loads[].loadPosition"的有效值, 以及动作参数"lhd"的有效值, 这些参数属于取放动作。 如果这个数组不存在或为空, 则 AGV 没有负载处理设备。
loadSets [loadSet]	array of JSON object	可由 AGV 处理的负载集数组
{		
setName	string	负载集的唯一名称。例如:DEFAULT、SET1 等
loadType	string	负载类型, 例如, EPAL、XLT1200 等
loadPositions	array of string	负载位置数组, 这个负载集是有效的。 如果此参数不存在或为空, 此负载集适用于该 AGV 上的所有负载处理设备。
boundingBoxReference	JSON object	状态消息中参数 loads[] 中定义的边界框引用。
loadDimensions	JSON object	状态消息中参数 loads[] 中定义的负载维度。
maxWeight	float64	[kg], 负载类型的最大重量。
minLoadhandlingHeight	float64	[m], 处理此负载类型的最小允许高度和重量参考 boundingBoxReference。

Field	data type	description
<i>maxLoadhandlingHeight</i>	float64	[m], 处理此负载类型的最大允许高度和重量 参考 boundingBoxReference.。
<i>minLoadhandlingDepth</i>	float64	[m], 此负载类型的最小允许深度和重量 参考 boundingBoxReference.
<i>maxLoadhandlingDepth</i>	float64	[m], 此负载类型的最大允许深度和重量参 考 boundingBoxReference.
<i>minLoadhandlingTilt</i>	float64	[rad], 此负载类型和重量的最小允许倾斜度。
<i>maxLoadhandlingTilt</i>	float64	[rad], 此负载类型和重量的最大允许倾斜度。
<i>agvSpeedLimit</i>	float64	[m/s], 此负载类型和重量的最大允许速度。
<i>agvAccelerationLimit</i>	float64	[m/s ²], 此负载类型和重量的最大允许加速度。
<i>agvDecelerationLimit</i>	float64	[m/s ²], 此负载类型和重量的最大允许减速。
<i>pickTime</i>	float64	[s], 大约是拾取货物的时间
<i>dropTime</i>	float64	[s], 大约是放下货物的时间
<i>description</i>	string	自由格式文本：负载处理集的描述。
}		

vehicleConfig

此 JSON 对象详细介绍了车辆上运行的软件和硬件版本，以及网络信息的简要摘要。

Field	data type	description
<i>versions[versionInfo]</i>	array of JSON object	包含软件和硬件信息的键值对对象数组。
<i>key</i>	string	所使用的软件/硬件版本的密钥。 (例如， 软件 版本)
<i>value</i>	string	与密钥对应的版本。 (例如， v1.12.4-beta)
}		
<i>network {</i>	JSON object	关于车辆网络连接的信息。车辆运行期间，不 得更新所列信息。
<i>dnsServers</i>	array of string	车辆使用的域名服务器 (DNS) 数组。
<i>ntpServers</i>	array of string	车辆的网络时间协议 (NTP) 服务器阵列。

Field	data type	description
<i>localIpAddress</i>	string	用于与 MQTT 经纪人通信的先验分配的 IP 地址。请注意，此 IP 地址不应在操作期间修改/更改。
<i>netmask</i>	string	网络配置中使用的子网掩码对应于本地 IP 地址。
<i>defaultGateway</i>	string	车辆使用的默认网关，对应于本地 IP 地址。
}		

7 Best practice 最佳实践

本节包含附加信息，有助于在协议逻辑的同时促进共同理解。

7.1 错误引用

如果由于错误指令导致错误，AGV 应在 errorReferences 字段中返回一个有意义的错误引用（参见第 6.10.6 状态消息的实现）。这可能包括以下信息：

- headerId
- 主题 (order or instantAction)
- 如果错误是由订单更新 orderId 引起的，则 orderId 和 orderUpdatelD，如果错误是由操作引起的。
- 如果错误是由错误的操作参数引起的，则参数列表

如果由于外部因素（例如，预期位置无负载）而无法完成操作，则应引用 orderId。

7.2 参数格式

错误、信息和操作的参数设计为 JSON 对象数组，包含键值对。

Field	data type	description
actionParameter {	JSON object	指示操作的动作参数，例如，deviceId、loadId、外部触发器。
key	string	参数的值
value}	One of:array, boolean, number, string, object	属于密钥的参数值。

动作“someAction”的动作参数示例，具有 stationType 和 loadType 的键值对：

```
"actionParameters": [ {"key": "stationType", "value": "floor"}, {"key": "weight", "value": 8.5}, {"key": "loadType", "value": "pallet_eu"} ]
```

使用“key”：“actualKey”、“value”：“actualValue”的拟议方案的原因是保持实现通用。

“实际值”可以是任何可能的 JSON 数据类型，例如浮点、布尔，甚至对象。

8 术语表

8.1 定义

Concept	Description
Free navigation AGV	<p>使用地图规划自身路径的车辆。 主控仅发送起始和目标坐标。 车辆将其路径发送到主控。 当与主控的连接中断时，车辆能够继续其旅程。 自由导航车辆可能被允许绕过局部障碍物。车辆本身也可能进行接收/分配位置的微调。</p>
Guided vehicles (physical or virtual)	<p>由主控发送路径的车辆。 路径计算在主控中进行。当与主控通信中断时，车辆终止已释放的节点和边（即“base”），然后停止。 引导车辆可能被允许绕过局部障碍物。车辆本身也可能进行接收/分配位置的微调。</p>
Central map	地图将集中保存在主控制中。这最初是创建的，然后被使用。

德国汽车工业协会（VDA）将 650 多家制造商和供应商汇聚一堂。成员开发和生产汽车和卡车、软件、拖车、上层建筑、公共汽车、零件和配件以及新的移动产品。

我们代表汽车行业利益，并支持现代、面向未来的多式联运出行，实现气候中和。VDA 代表其成员在政治、媒体和社会团体中的利益。

我们致力于电动出行、气候友好型驱动、气候目标的实施、原材料保障、数字化和网络化以及德国工程。我们致力于打造具有竞争力的商业和创新中心。我们的行业确保了德国的繁荣：超过 78 万人直接在德国汽车行业就业。

VDA 是大型国际出行平台 IAA MOBILITY 的主办方，也是商用车行业未来最重要的平台 IAA TRANSPORTATION 的主办方。

Publisher German Association of the Automotive Industry
Behrenstraße 35, 10117 Berlin www.vda.de/en

German Bundestag Lobby Register No.: R001243
EU Transparency Register No.: 9557 4664 768-90

Copyright German Association of the Automotive Industry
Reprint, also in extracts, is only permitted,
if the source is stated.

Version Version 2.1.0, January 2025

Verband der Automobilindustrie