基于构件的软件架构设计——应急监控系统

一、实验目的及要求

通过实践一个实时系统“自动引导车辆系统”的需求建模、需求分析、架构设计、详细设计的全过程，掌握并发和实时软件架构分析和设计方法。熟练使用 UML 语言及其相关建模方法，熟练使用 StartUML 建模工具进行相关设计开发。

二、实验设备（环境）及要求

1．PC 机最低配置：2G Hz 以上 CPU；1G 以上内存；1G 自由硬盘空间

2．StartUML

三、实验内容与步骤

1．安装 StartUML；

2．对“自动引导车辆系统”开发用例模型，详细了解用例编档的方法；

3．面向问题域的静态建模；

4．识别系统中的各类对象；

5．动态建模，着重考察动态状态机建模方法；

6．实践并发软件架构设计；

7．理解并发通信模式；

8．设计并发任务及任务接口。

四、实验结果与数据处理

注：本实验使用 ProcessOn 进行绘图。

1 问题描述

基于计算机的自动车辆引导系统（AGV）可以在工厂内部按照顺时针方向在轨道上移动，并且在工厂内各个站点启动和停止。AGV 系统具有以下特点:

1）一个发动机，可以按照命令启动和停止车辆的移动。发动机会发送两种响应:“已启动”(Started）和“已停止”(Stopped)。

2）一个到达传感器，用于检测 AGV 何时到达一个站点，例如已到达 x 站点。如果该站点是目标站点，则 AGV 会停止。反之 AGV 会越过该站点继续移动。

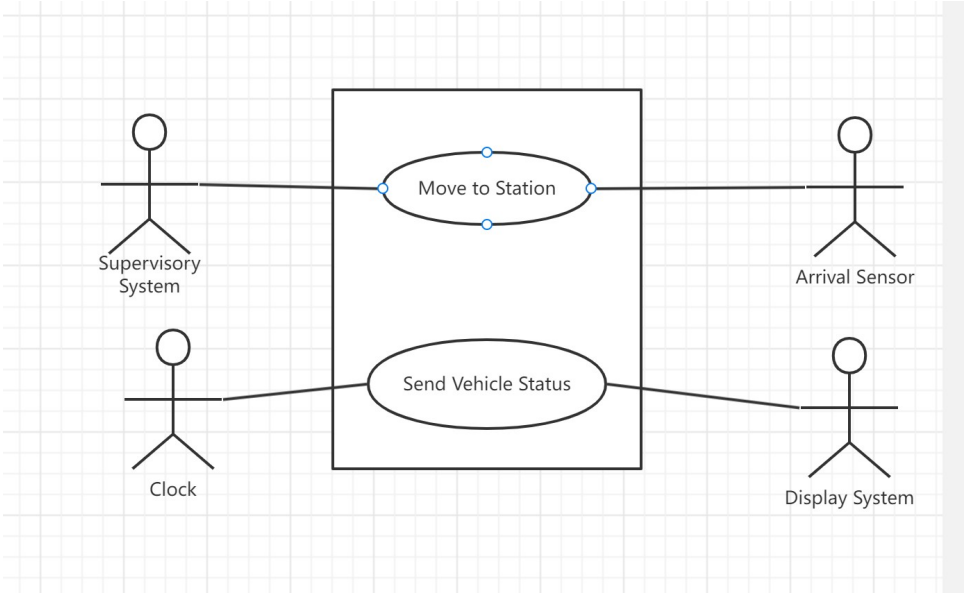
3）一个机械臂,用于装货和卸货。

AGV 系统从外部“监管系统”那里接收“移动”(Move)命令，并且向“监管系统”发送车辆“确认信息”(Acks)，以表明车辆已启动、越过某个站点或在某站点上停止。此外,AGV 系统还会每隔 30 秒向“显示系统”发送一次车辆状态信息。

在 AGV 系统中到达传感器是事件驱动的输人设备，而发动机和机械臂则是被动的输入/输出（IO）设备。此外，AGV 系统通过消息与“监管系统”和“显示系统”进行通信。

2 用例建模

用例模型如图所示：



用例描述如下所示。

2.1 “移动到站点”用例

用例名称:移动到站点

概述:AGV 移动部件到工厂站点

参与者:监管系统（主)，到达传感器（次)

前置条件:AGV 是静止的

主序列:

1.“监管系统”发送消息给“AGV 系统”，要求移动到某个工厂站点并装载一个部件。

2."AGV 系统”命令发动机开始移动。

3.发动机通知“AGV 系统”车辆已经开始移动。

4.“AGV 系统”发送“已离开”的消息到“监管系统”。

5.到达传感器通知“AGV 系统”已经到达.工厂站点（#)。

6.“AGV 系统”确定该站点是目标站点并命令发动机停止移动。

7.发动机通知“AGV 系统”车辆已经停止移动。

8.“AGV 系统”命令机械臂装载部件。

9.机械臂通知“AGV 系统”部件已经装载。

10."AGV 系统”发送“已到达”(Arrived）消息给“监管系统”。

可替换序列:

第 6 步:如果车辆到达与目标站点不同的站点，车辆将继续移动越过该站点并发送消息“越过工厂站点（#)”给“监管系统”。

第 8、9 步:如果“监管系统”请求 AGV 移动到工厂站点并卸载部件，AGV 将会在到达目标站点后卸载部件。

后置条件:AGV 完成任务并到达目标站点。

2.2 “发送车辆状态”用例

用例描述如下：

用例名称:发送车辆状态

概述:AGV 发送关于车辆位置和空闲/繁忙状态的信息给显示系统。

参与者:时钟（主)、显示系统（次）

前置条件:AGV 处于运行状态

主序列:

1.时钟通知“AGV 系统”计时器已过期。

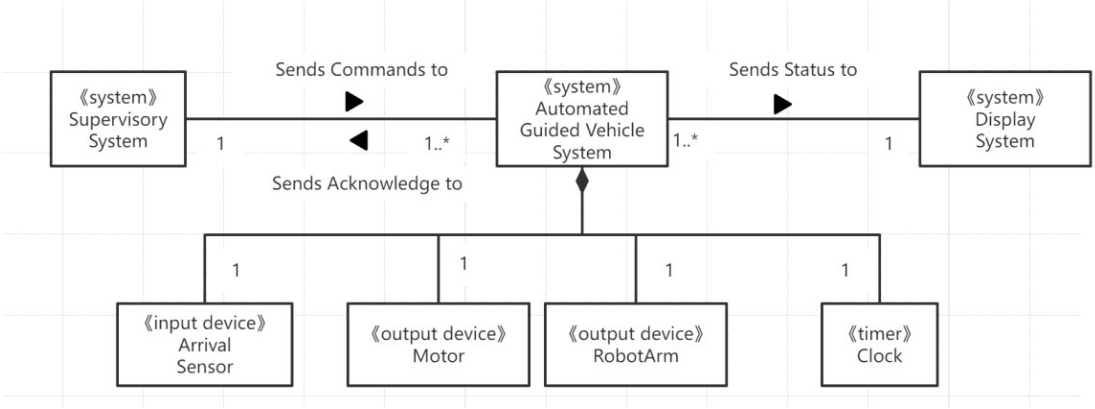
2.“AGV 系统”读取关于 AGV 位置和空闲/繁忙状态的信息。

3.“AGV 系统”发送其状态信息给“显示系统”。

后置条件:“AGV 系统”已经发送状态信息

3 静态建模

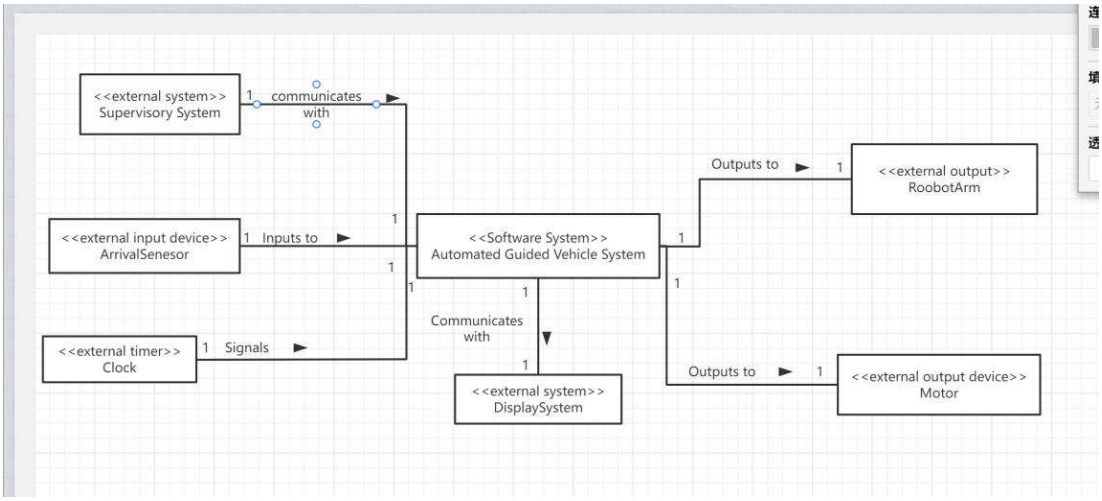
3.1 概念静态模型



如图所示。图中给出了用类图表示的概念静态模型。

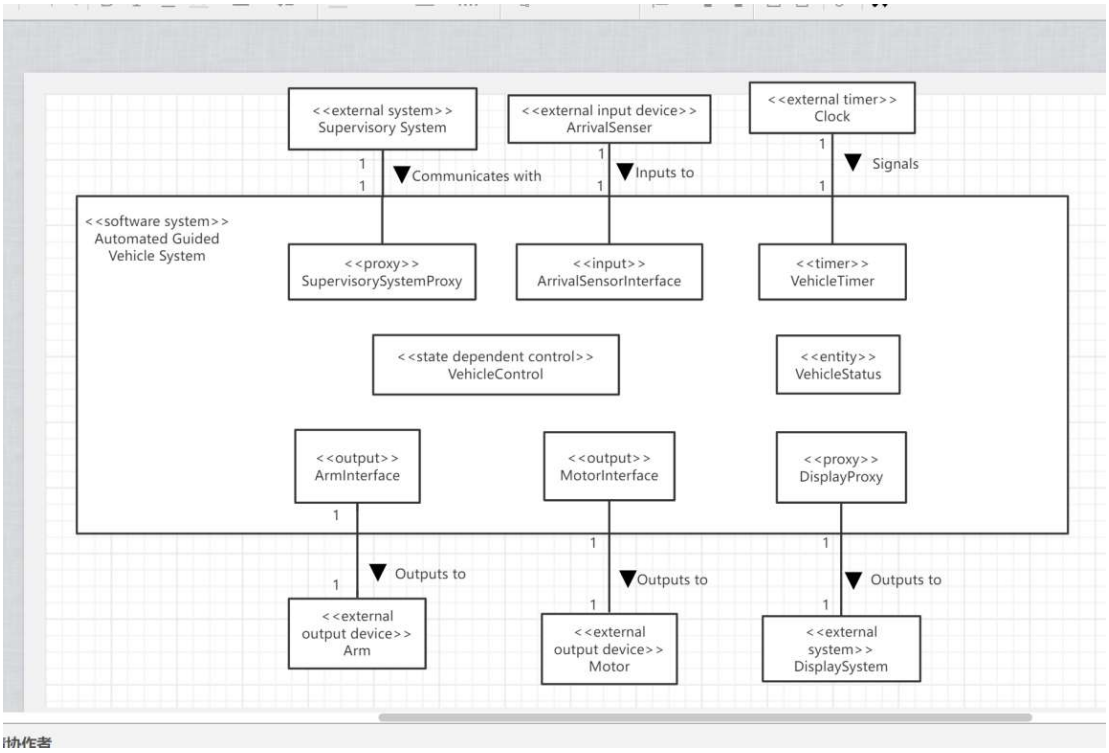
3.2 软件系统上下文建模

“自动引导车辆系统”的软件系统上下文类图如下图所示：



软件系统上下文类图是从被开发软件系统（即“AGV 系统”）的角度建模的。因此，它包括两个外部系统类（即“监管系统”和“显示系统”）以及作为外部计时器类的“时钟”，“时钟”最初在用例模型中被描述为参与者。此外还有一个外部输入设备类“到达传感器”和两个外部输出设备类“发动机”和“机械臂”。

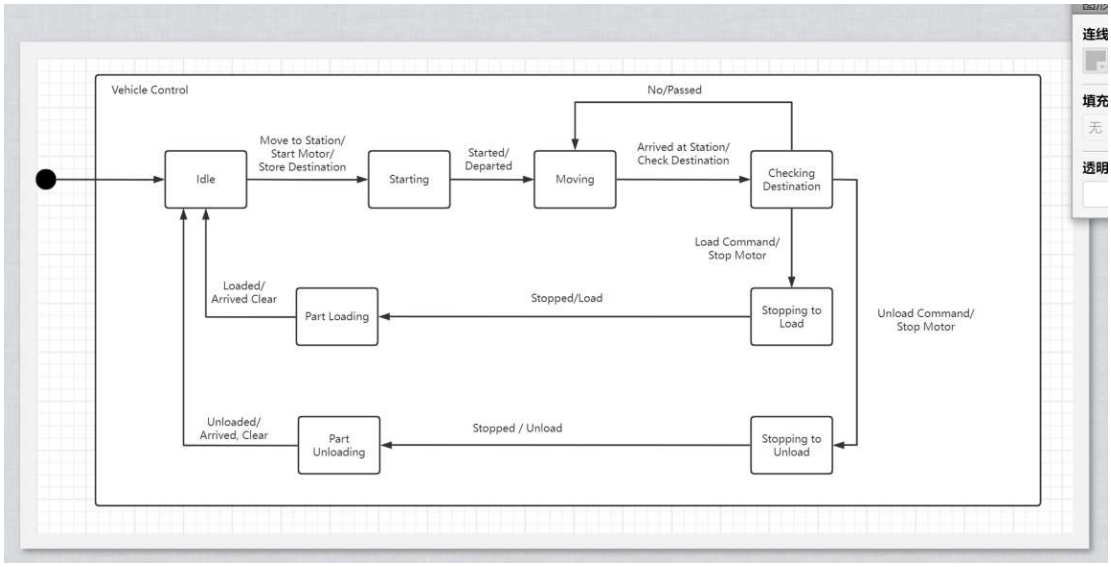
4 对象和类组织



“AGV 系统”的对象组织如图所示。软件系统上下文类图中的每个外部类都有个相对应的内部软件类。因此，存在两个代理类，即“监管系统代理”(Supervisory SystemProxy）和“显示代理”(Display Proxy)，它们分别与两个外部系统通信，即“监管系统”(Supervisory System)和“显示系统”（Display System)。还有一个输人类“到达传感器接口”(Arrival Sensor Interface)，它与外部输入设备“到达传感器”(Arrival Sensor)进行通信；有两个输出类“发动机接口”(Motor Interface）和“机械臂接口”(Arm Interface)，分别与两个外部输出设备类“发动机”(Motor ）和“机械臂”(Arm)通信。此外，系统中还存在两个类:一个状态相关的控制类“车辆控制”(Vehicle Control)，负责执行车辆状态机;一个实体类“车辆状态”(Vehicle Status)，其中包含了关于车辆目的地和命令的数据。最后，还存在一个计时器类“车辆计时器”(Vehicle Timer)。

5 动态状态机建模

“车辆控制”类执行车辆状态机，如图中的状态图所示。



状态机涵盖了车辆从空闲(idle ）状态到移动、到达目的地、装载或卸载部件以及重新启动等各种状态。这些状态可以通过分析“移动到站点”用例中所描述的主序列来确定，具体如下:

空闲（Idle)。初始状态，此时 AGV 空闲并等待来自“监管系统”的命令。

启动(Starting)。当 AGV 接收到来自“监管系统”的“移动到站点”(MovetoStation)消息并发送一个启动命令给发动机后进入此状态。

移动(moving)。AGV 正在向下一个站点移动。

检查目的地（Checking Destination)。AGV 到达一个站点并正在检查以确定该站点是否是目标站点。

停止并准备装载(Stopping to Load)。当前站点为目标站点，AGV 准备装载部件。AGV 在进入此状态时命令发动机停止。

部件装载（Part Loading)。机械臂正在将部件装载到 AGV 上。

停止并准备卸载(Stopping to Unload)。当前站点为目标站点，AGV 准备卸载部件。AGV 在进入此状态时命令发动机停止。

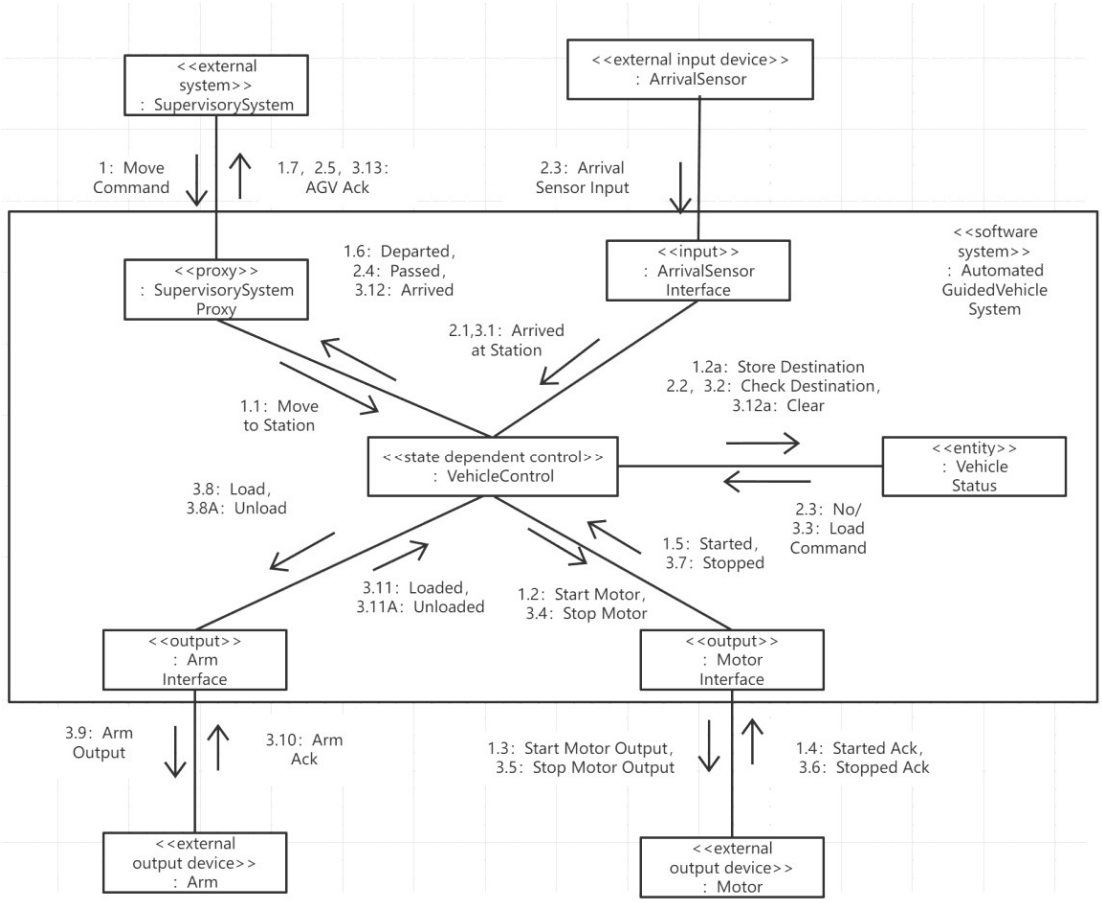
部件卸载（Part Unloading)。机械臂正在将部件从 AGV 上卸载下来。

6 动态交互建模

对于每个用例，我们都要开发一个通信图以描述参与用例的各个对象以及对象之间传递的消息序列。其中主要的用例“移动到站点”(Move to Station）由多个对象共同实现,而支持用例“发送车辆状态”(Send Vehicle Status）只需要三个软件对象来实现。

6.1 “移动到站点”用例的动态建模

该用例的通信图如下图所示：



车辆经过第一个站点，并且停在第二个站点以上装载部件，消息序列如下：

1:外部“监管系统”向“AGV 系统”发送“移动”命令，要求车辆移动到一

个工厂站点并装载部件。

1.1:“监管系统代理”接收“移动”命令，发送“移动到站点”命令给“车辆控制”。

1.2:“车辆控制”发送“启动发动机”命令给“发动机接口”以开始移动。

1.2a:“车辆控制”在“车辆状态”中保存目标站点以及装载/卸载命令。

1.3:“发动机接口”发送“启动发动机”(Start Motor)命令给外部“发动机”。

1.4:“发动机”发送“启动”确认信息给“发动机接口”。

1.5:“发动机接口”通知“车辆控制”车辆已经开始移动。

1.6:“车辆控制”发送“离站”消息给“监管系统代理”。

1.7:“监管系统代理”将“离站”(Departed）消息转发给“监管系统”

2:到达传感器通知 AGV 系统已经到达工厂站点（#)。

2.1:“到达传感器接口”发送“到达站点”(Arrived at Station）消息给“车辆控制”。

2.2:“车辆控制”发送“检查目的地”(Check Destination）消息给“车辆状态”。

2.3:“车辆状态”对象指示此站点不是目的地站点。

2.4:“车辆控制”发送“继续移动”(Passed）的消息给“监管系统代理。”

2.5:“监管系统代理”将“继续移动”的消息转发给“监管系统”。

3:到达传感器通知“AGV 系统”它已经到达工厂站点（#)。

3.1:“到达传感器接口”发送“到达站点”消息给“车辆控制”。

3.2:“车辆控制”发送“检查目标站点”消息给“车辆状态”对象。

3.3:“车辆状态”对象指示此站点为目标站点，并表明所接到的命令是装载

部件。

3.4:“车辆控制”发送“停止发动机”消息给“发动机接口”。

3.5:“发动机接口”发送“停止发动机”命令给外部“发动机”设备。

3.6:“发动机”发送“停止”确认消息给“发动机接口”。

3.7:“发动机接口”通知“车辆控制”车辆已经停止移动。

3.8:“车辆控制”发送“装载”(Load）消息给“机械臂接口”。

3.9:“机械臂接口”发送“装载”消息给外部“机械臂”设备。

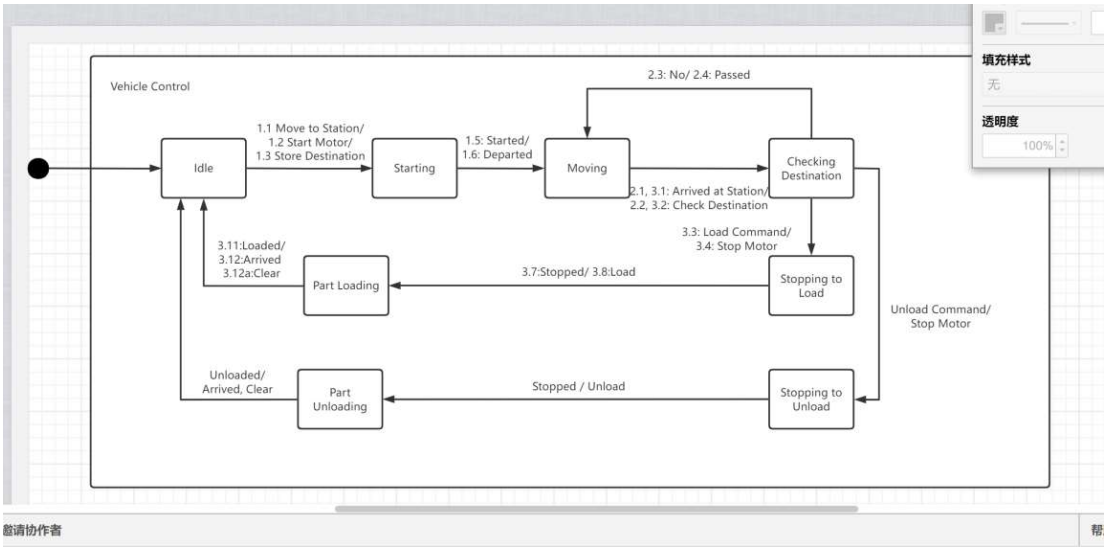
3.10:“机械臂”发送确认消息给“车辆控制”并表明机械臂已完成装载。

3.11:“机械臂接口”发送“已装载”（ Loaded ）消息给“车辆控制”。

3.12:“车辆控制”发送“到达”消息给“监管系统代理”。

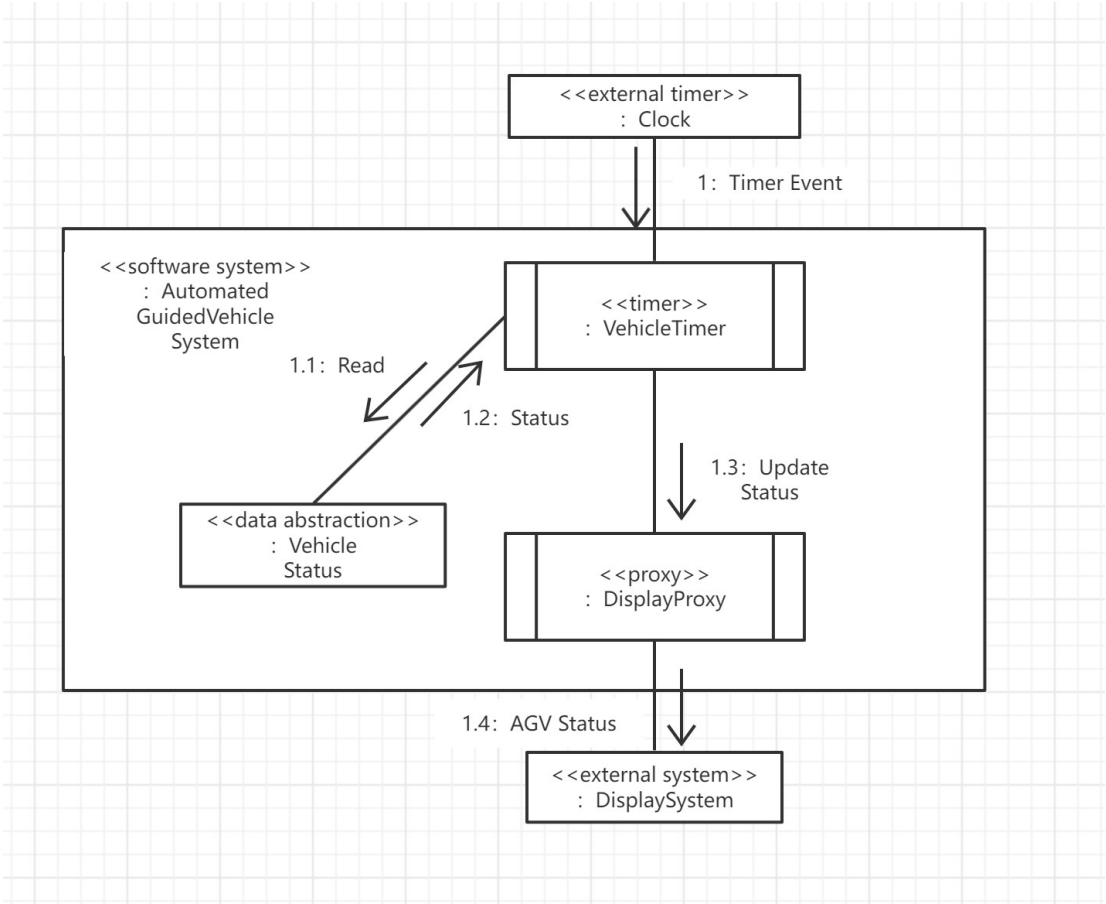
3.13:“监管系统代理”转发“到达”消息给“监管系统”。

“车辆控制”发出和接收的消息与状态图中的事件和动作相对应，并且遵循与此前的消息序列描述中一样的场景。



6.2 “发送车辆状态”用例的动态建模

通信图如下图所示：



消息序列如下：

1:“时钟”发送“计时器事件”给“车辆计时器”。

1.1，1.2:“车辆计时器”读取“车辆状态”。

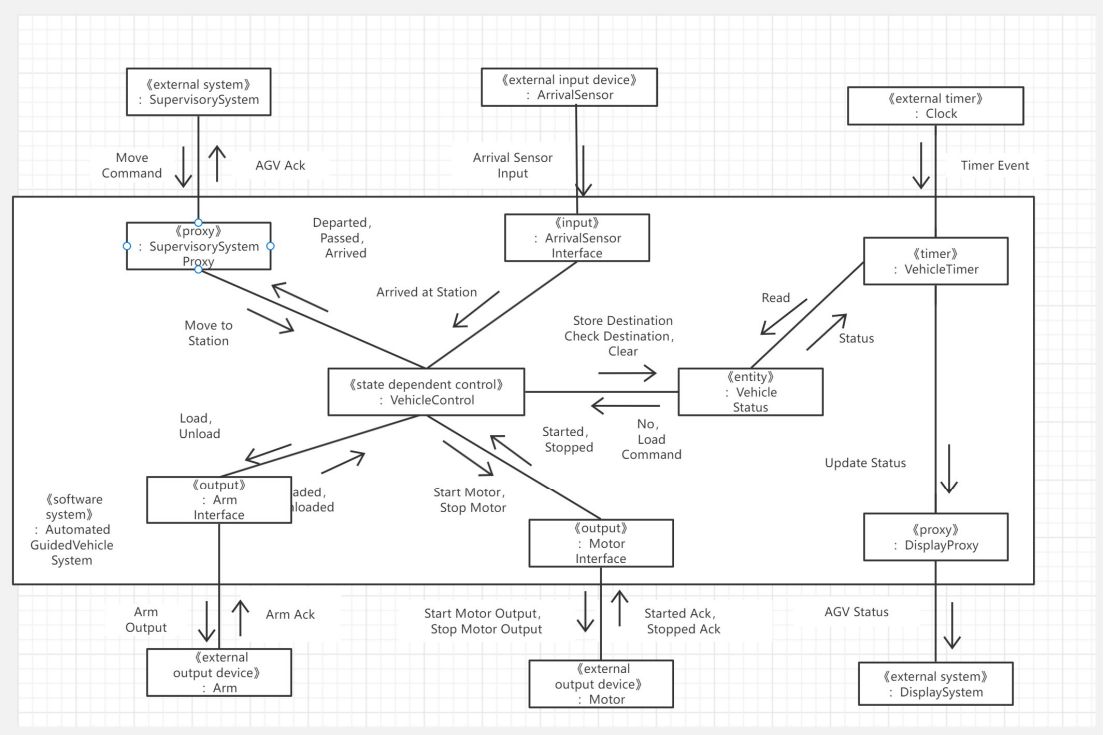
1.3:“车辆计时器”发送“更新状态”消息给“显示代理”。

1.4:“显示代理”发送“车辆状态”给外部“显示系统”。

7 设计建模

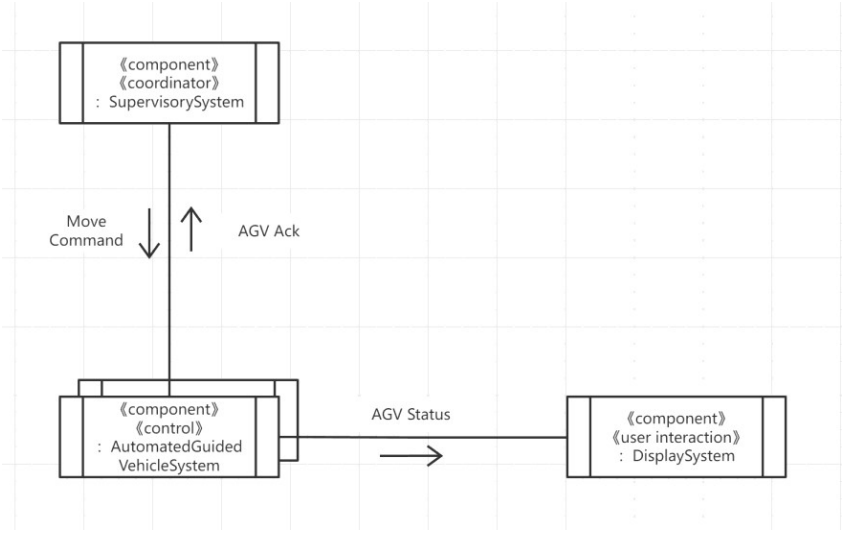
7.1 集成通信图

如图所示。

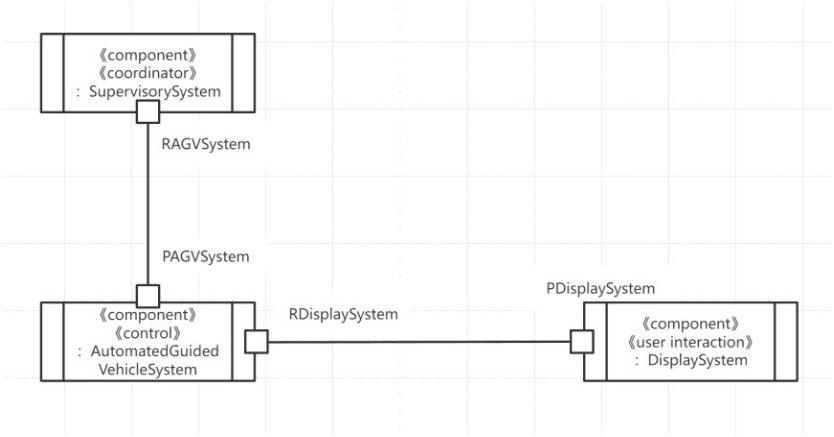


图中的集成非常简单，因为参与两个基于用例的通信图中只有一个共有对象，即“车辆状态”（ Vehicle Status)。集成通信图是种用于描述各个对象之间所有可能通信的泛化的通信图。

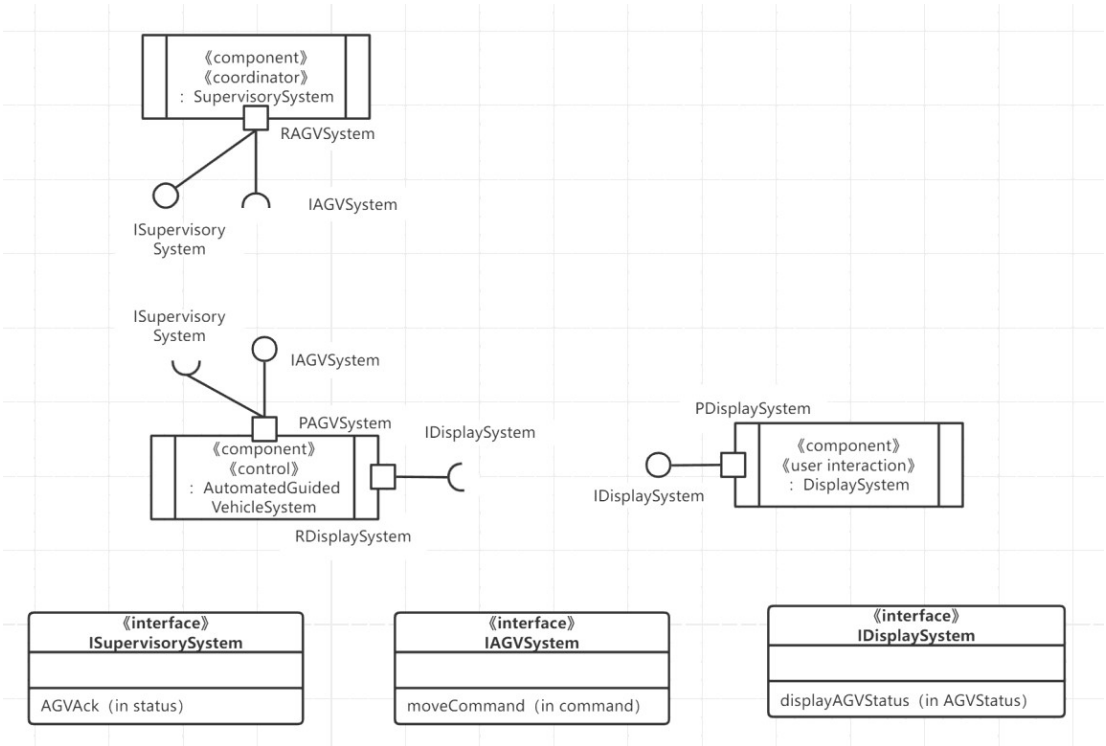
7.2 基于构件的工厂自动化系统软件体系结构



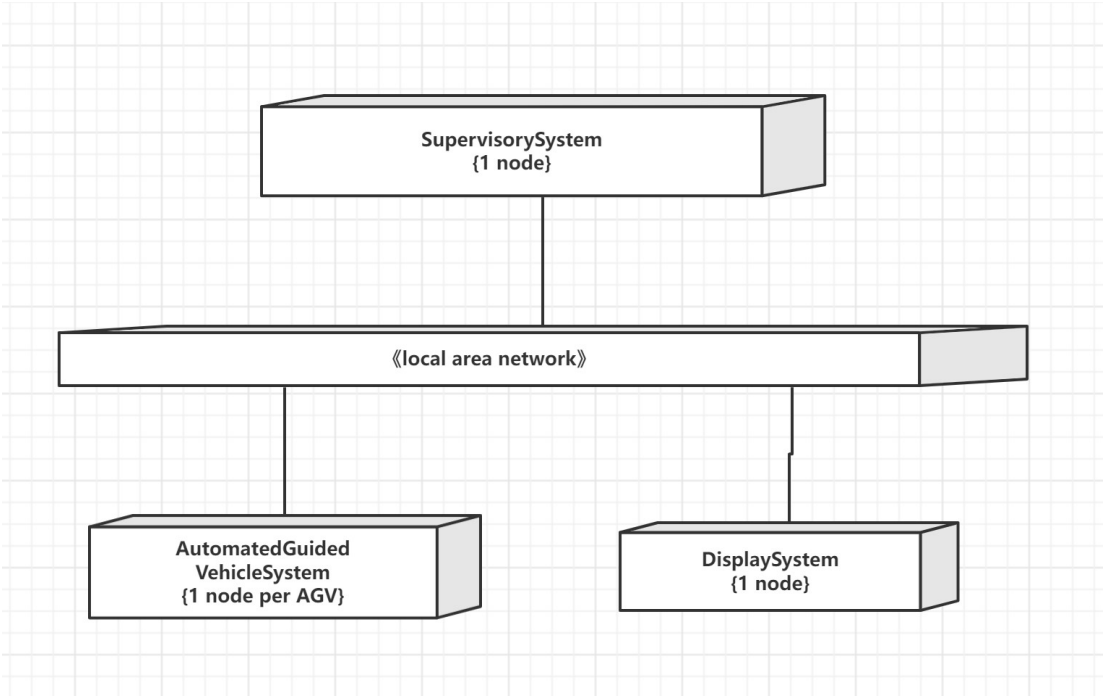
上图中显示了三个相交互的分布式系统（设计为构件):“监管系统”(SupervisorySystem)、“自动引导车辆系统”(AGV)、“显示系统”(Display System)。其中包含一个“监管系统”的实例、多个“AGV 系统”和“显示系统”的实例。所有分布式构件之间的通信均为异步的，这为消息通信提供了极大的灵活性。“监管系统”和“AGV 系统”之间的通信是一个双向异步通信的例子。



上图描述了基于构建的“工厂自动化系统”的软件体系结构。其复合构件端口和接口如下图所示：



“工厂自动化系统”的分布式系统部署如下图所示：



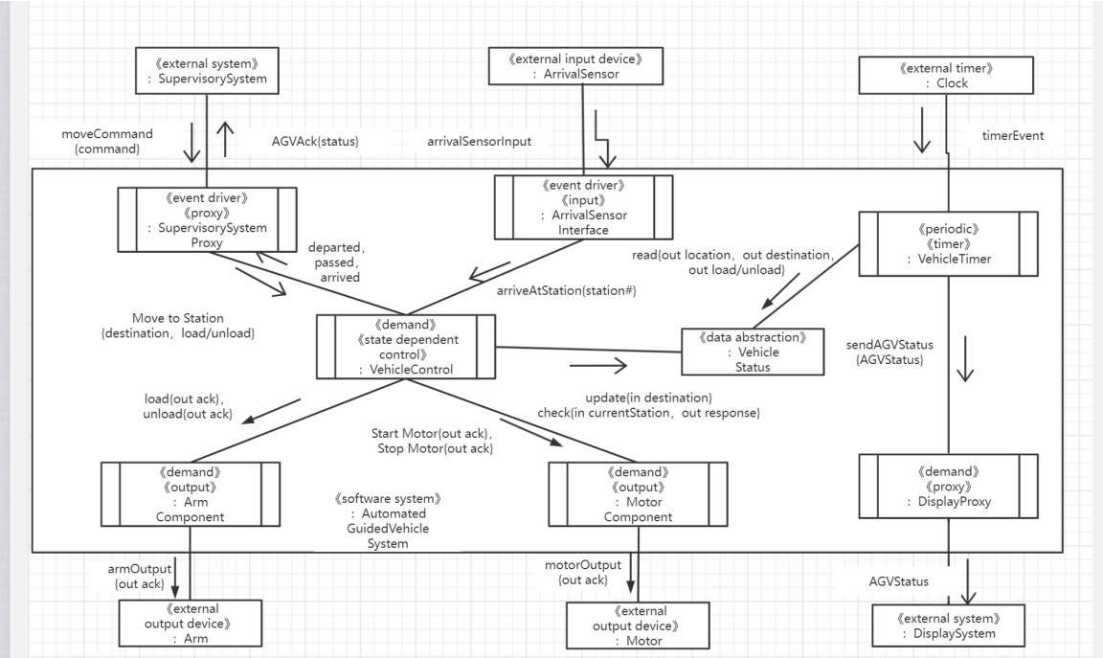
7.3 自动引导车辆系统的软件体系结构

“AGV 系统”的设计基于实时设计的集中式控制模式，其中一个控制构件“车辆控制”（ Vehicle Control）提供对整个系统的控制。此外，“AGV 系统”被设计成基于构件的分布式软件体系结构，这种设计允许输入输出构件驻留在不同的结点上，而这些结点之间通过高速总线相连。在系统部署时需要确定所采用的配置类型（集中式或分布式)。

“AGV 系统”的并发软件体系结构是在集成通信图基础上开发的。其中，并发任务是通过应用任务结构化组织准则设计的，而任务之间的消息通信是通过应用体系结构通信模式设计的。随后设计基于构件的软件体系结构。最后描述每个构件的供给接口和请求接口。每个构件端口则在供给接口和/或请求接口之上定义。

7.4 并发软件体系结构

并发通信图如下图所示：



并发任务如下所示：

输入任务。并发输入任务从外部环境接收输人并发送相应消息给控制任务。“到达传感器构件”被设计为事件驱动的输入任务，会被到来的到达传感器输人唤醒。输入任务由分析模型中各个输入设备接口对象组成:“到达传感器接口”。

代理任务。“监管系统代理”代表“监管系统”运行，从“监管系统”那里接收“移动”命令并转发给“车辆控制”任务，并且向“监管系统”发送 AGV 确认消息。“监管系统代理”被设计成事件驱动任务，由来自外部“监管系统”或内部“车辆控制”的消息唤醒。值得注意的是，如果一个任务不仅接收外部消息而且还接收内部消息,则该任务被设计成事件驱动任务而非按需驱动任务。“显示代理”(Display Proxy)在“显示系统”一端执行操作，负责将 AGV 状态消息转发给“显示系统”。“显示代理”被设计成按需驱动任务，由来自于“车辆计时器”的消息唤醒。

控制任务。“车辆控制”任务是“AGV 系统”的一个集中式状态相关的控制任务。该任务执行“车辆控制”状态机，从其他包含能够引起“车辆控制”状态变化的事件的任务接收消息，并且发送动作消息给其他任务。“车辆控制”任务被设计成按需驱动任务，由来自于“监管系统代理”和“到达传感器构

件”的消息唤醒。

输出任务。“机械臂构件”与外部“机械臂”项链。在分析模型中“机械臂接口”对象被映射为输出任务。与之相似，“发动机构件”与外部“发动机”交互，在分析模型中发动机构件被设计为“发动机接口”对象。两个输出任务都被设计为按需驱动任务，由来自于“车辆控制”任务的消息按需唤醒。

7.5 体系结构通信模式

该系统有如下四种通信模式：

1）异步消息通信。“异步消息通信”模式在“AGV 系统”中得到了广泛使用，这是因为系统中的大部分通信是单向的，而且此模式的一个优势是消息生产者无需为消息消费者挂起。“车辆控制”任务需要从两个消息生产者（即“监管系统代理”和“到达传感器构件”)接收消息而不论其顺序。为了灵活性，处理此需求的最佳方法是通过异步消息通信，其中为“车辆控制”任务建立一个输入消息队列，从而使“车辆控制”任务可以处理首先到达的消息，不论是移动命令还是到站通知。“车辆计时器”任务发送异步的 AGV 状态消息给“显示代理”任务，而“显示代理”同样通过消息队列接收这些消息。

2）双向异步通信。此通信模式用在“监管系统代理”与“车辆控制”任务之间，这是由于在“监管系统代理”发送移动命令给“车辆控制”任务与“车辆控制”任务发回确认消息之间可能需要相当长的一段时间（确认发生在 AGV 到达目标站点之后)。因此，移动命令和确认消息之间不存在耦合。

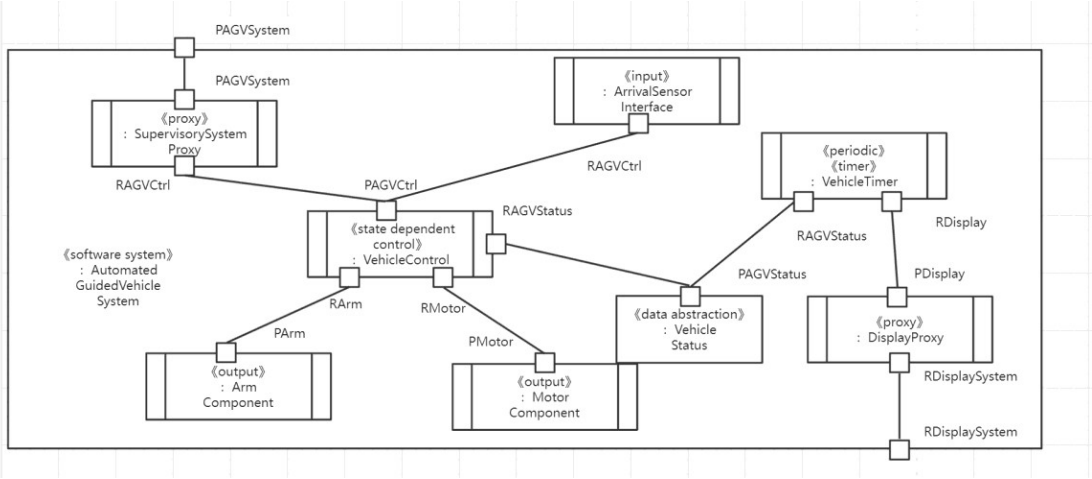
3）不带回复的同步消息通信。当消息生产者需要在继续执行之前确认消息消费者已经接收到消息时需要使用此模式。在本章的实例中，此通信模式用在“车辆控制”和“机械臂构件”以及“车辆控制”和“发动机构件”之间。这两种情形下消费者任务会保持空闲直到接收到消息，因此消息生产者“车辆控制”任务发送消息后无需等待确认。

4）调用/返回。当“AGV 控制”和“车辆计时器”调用被动“车辆状态”数

据抽象对象时使用。

7.6 基于构建的软件体系结构

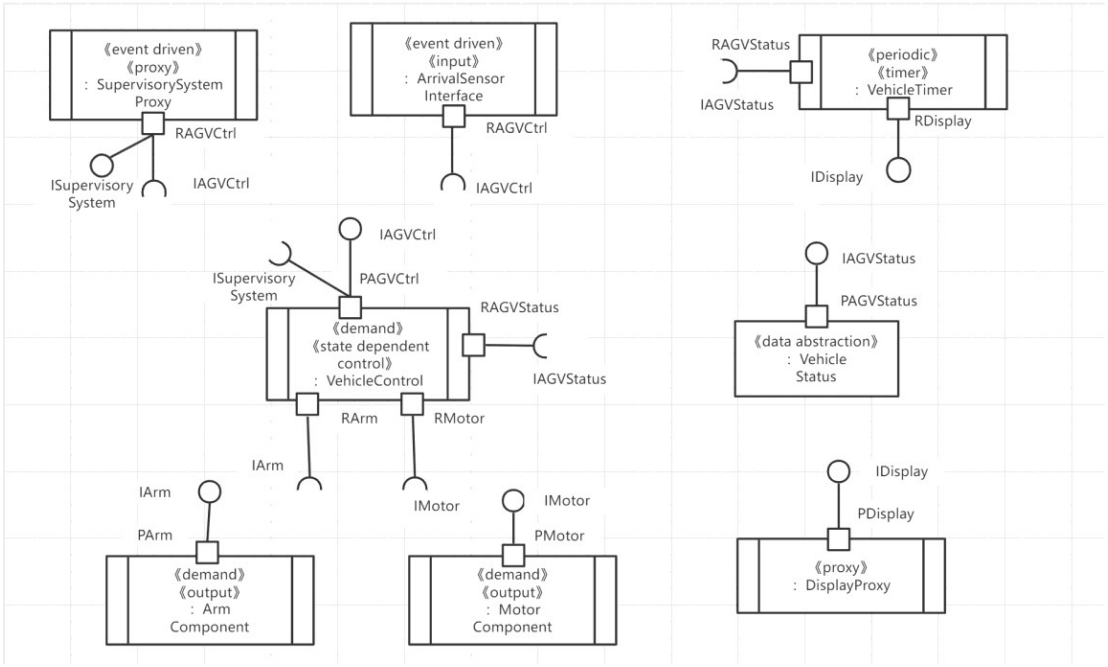
基于构件的“AGV 系统”体系结构如图所示。



图中显示了一个包含系统构件端口和连接器的 UML 复合结构图。除一个构件之外所有其他构件都是并发的，构件之间通过端口进行通信。系统的整体结构以及构件之间的连接在“AGV 系统”的并发通信图的基础上确定。因此，构件系结构的复合结构图是基于并发通信图确定的。

7.7 构件接口设计

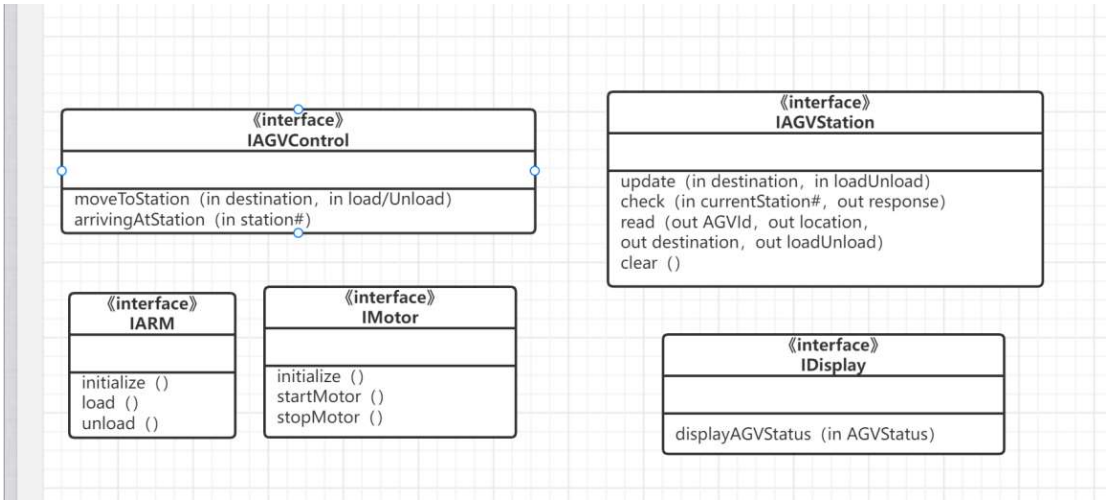
每个构件端口都是按照其请求接口和/或供给接口进行定义的。有些消息生产者构件(特别是输入构件）不提供软件接口，这是由于此类构件直接从外部硬件输入设备接收输入，但是这些构件需要通过控制构件提供的接口向控制构件发送消息。



上图给出了输人构件“到达传感器构件”(Arrival Sensor Component）的端口和请求接口。此输人构件以及“监管系统代理”构件具有相同的请求接口IAGVControl，该接口由“车辆控制”构件提供。

“车辆控制”构件具有三个请求端口，通过这些端口“车辆控制”构件可以发送消息给两个输出构件(“机械臂构件”和“发动机构件”)的供给端口，而且“车辆控制”构件通过“车辆状态”数据抽象对象的供给端口调用其操作。

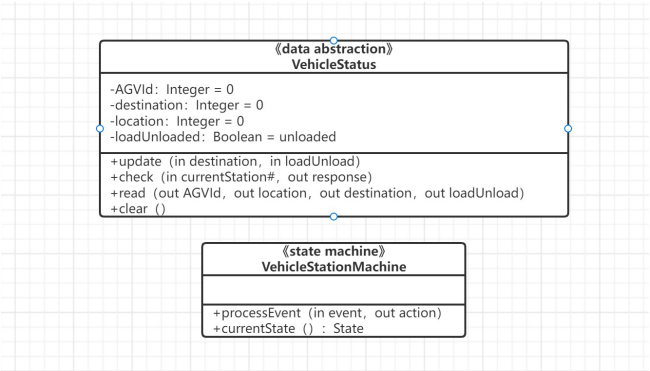
输出构件因为直接输出至外部硬件输出设备而无需软件接口，但是需要提供接口以接收来自于控制构件的消息。上图给出了“AGV 系统”中两个输出构件的端口和供给接口。



上图给出了这两个输出构件接口的规约，规约中说明了它们提供的操作。“机械臂构件”和“发动机构件”输出构件均具有一个供给端口:

“机械臂构件”的 PArm 端口，提供接口 IArm;“发动机构件”的 PMotor 端口，提供接口 IMotor。

“车辆控制”构件执行 AGV 状态表，并且从两个生产者构件接收异步控制请求消息。供给接口 IAGVControl 在上图中说明，接口很简单，只包含一个操作 processControlRequest，该操作具有一个输入参数 controlRequest，参数中包含单个消息的名称和内容。考虑到系统的演化，在接口中为每个控制请求分别设计操作会使接口过于复杂，因为系统演化通常需要操作的增加或删除而不仅仅是参数的变更。



上图显示了被动数据抽象对象“车辆状态”的各个属性。图中给出了名为“车辆状态机”设计的状态机类，该类被封装在“车辆控制”构件中。

五、分析与讨论

1．状态相关控制对象封装的状态机使用 E-C-A 机制来执行，请问这种执行机制在系统的集成通信图中怎样实现状态相关控制对象与其他对象之间的通信协作？

“AGV 系统”的设计基于实时设计的集中式控制模式，其中一个控制构仁“车辆控制”(Vehicle Control）提供对整个系统的控制。此外，“AGV 系统”被设计成基于兮件的分布式软件体系结构，这种设计允许输入输出构件驻留在不同的结点上，而这些结点之间通过高速总线相连。在系统部署时需要确定所采用的配置类型(集中式或分布式)。“AGV 系统”的并发软件体系结构是在集成通信图基础上开发的。其中，并发任务是通过应用任务结构化组织准则设计的，而任务之间的消息通信是通过应用体系结构通信模式设计的。随后设计基于构件的软件体系结构。最后描述每个构件的供给接口和请求接口。每个构件端口则在供给接口和/或请求接口之上定义。

2．并发实时系统中涉及到哪些消息通信模式？举例说明。

有如下四种通信模式：

1）**异步消息通信。**“异步消息通信”模式在“AGV 系统”中得到了广泛使用，这是因为系统中的大部分通信是单向的，而且此模式的一个优势是消息生产者无需为消息消费者挂起。“车辆控制”任务需要从两个消息生产者（即“监管系统代理”和“到达传感器构件”)接收消息而不论其顺序。为了灵活性，处理此需求的最佳方法是通过异步消息通信，其中为“车辆控制”任务建立一个输入消息队列，从而使“车辆控制”任务可以处理首先到达的消息，不论是移动命令还是到站通知。“车辆计时器”任务发送异步的 AGV 状态消息给“显示代理”任务，而“显示代理”同样通过消息队列接收这些消息。

2）**双向异步通信。**此通信模式用在“监管系统代理”与“车辆控制”任务之间，这是由于在“监管系统代理”发送移动命令给“车辆控制”任务与“车

辆控制”任务发回确认消息之间可能需要相当长的一段时间（确认发生在 AGV 到达目标站点之后)。因此，移动命令和确认消息之间不存在耦合。

3）不带回复的同步消息通信。当消息生产者需要在继续执行之前确认消息消费者已经接收到消息时需要使用此模式。在本章的实例中，此通信模式用在“车辆控制”和“机械臂构件”以及“车辆控制”和“发动机构件”之间。这两种情形下消费者任务会保持空闲直到接收到消息，因此消息生产者“车辆控制”任务发送消息后无需等待确认。

4）调用/返回。当“AGV 控制”和“车辆计时器”调用被动“车辆状态”数据抽象对象时使用。

六、教师评语 签名： 日期：	成绩