A feature-based ontology for cyber-physical systems

Bedir Tekinerdogan, Mittal Rakshit, Rima Al Ali, Mauro Iacono, Eva Navarro-López, Soumyadip Bandyopadyay, Ken Vanherpen, Ankica Barisic, Kuldar Taveter

Chapter 3

A Feature-Based Ontology for Cyber-Physical Systems

Author(s): Bedir Tekinerdogan, Rakshit Mittal, Rima Al-Ali, Mauro Iacono, Eva Navarro, Soumyadip Bandyopadyay, Ken Vanherpen, Ankica Barišić, Kuldar Taveter

3.1 Introduction

Cyber-Physical Systems (CPS) are systems that tightly integrate computation with networking and physical processes. Such systems form large networks that communicate with each other and rely on actuators and sensors to monitor and control complex with physical processes, creating complex feedback loops between the physical and the cyberworlds. CPS bring innovation in terms of economic and societal impacts for various kinds of industries, creating entirely new markets and platforms for growth. CPS have growing applications in various domains, including healthcare, transportation, precision agriculture, energy conservation, environmental control, avionics, critical infrastructure control (electric and nuclear power plants, water resources, and communications systems), high confidence medical devices and systems, traffic control and safety, advanced automotive systems, process control, distributed robotics (telepresence, telemedicine), manufacturing, and smart city engineering. The positive economic impact of any one of these applications areas is enormous.

Technically, CPS systems are inherently heterogeneous, typically comprising mechanical, hydraulic, material, electrical, electronic, and computational components. The engineering process of CPS requires distinct disciplines to be employed, resulting in a collection of models that are expressed using correspondingly distinct modeling formalisms. An important realization is that distinct models need to be weaved together consistently to form a complete representation of a system that enables, among other global aspects, performance analysis, exhaustive simulation and verification, hardware in the loop

simulation, determining best overall parameters of the system, prototyping, or implementation.

A new framework is required that is able to represent these connections between models and, moreover, enable reasoning about them. No single formalism is able to model all aspects of a system; modeling of a CPS system is inherently multi-paradigm, which calls for a trans-disciplinary approach to be able to conjoin abstractions and models from different worlds. Physically, CPS systems are inherently heterogeneous, typically comprising mechanical, hydraulic, material, electrical, electronic, among others. Those areas correspond to engineering disciplines with their own models and abstractions designed to best capture the dynamics of physical processes (e.g., differential equations, stochastic processes, etc.). Computationally, CPS systems leverage the half-century old knowledge in computer science and software engineering to essentially capture how data is transformed into other useful data, abstracting away from core physical properties occurring in the real world, and particularly the passage of time in physical processes.

The key challenge, as identified a decade ago, is then to provide mathematical and technical foundations to conjoin physical abstractions that describe the dynamics of nature in various engineering domains, as described earlier, with models focusing solely on data transformation. This is necessary to adequately capture and bridge both aspects of a complex, realistic cyber-physical system, and become able to reason and explore system designs collaboratively, allocating responsibilities to software and physical elements, and analyzing trade-offs between them.

This chapter aims to provide an ontology for CPS to support multi-paradigm modeling for CPS. To this end we will primarily use metamodeling and domain analysis. The result of the metamodeling activity will be a metamodel consisting of all the relevant concepts and their relations. For the domain analysis process we will adopt a feature model to represent both the common and variant features of the CPS domain.

The remainder of this chapter is organized as follows. In section 2 we present the overall approach for providing the ontology. Section 3 presents the metamodel and feature diagram for CPS as a result of the domain analysis process. Section 4 describes the CPS architecture that builds on the feature model and metamodel. Finally, section 5 provides the conclusion

3.2 Metamodel of Cyber-Physical Systems

The domain analysis process has resulted in a set of concepts of CPS which have been represented in a metamodel as shown in Figure 3.1 [39],[40]. The metamodel is a complementary model to a feature diagram and is used to show the relationships among the concepts. The feature diagram on the other hand stresses the commonality and variability of the CPS features.

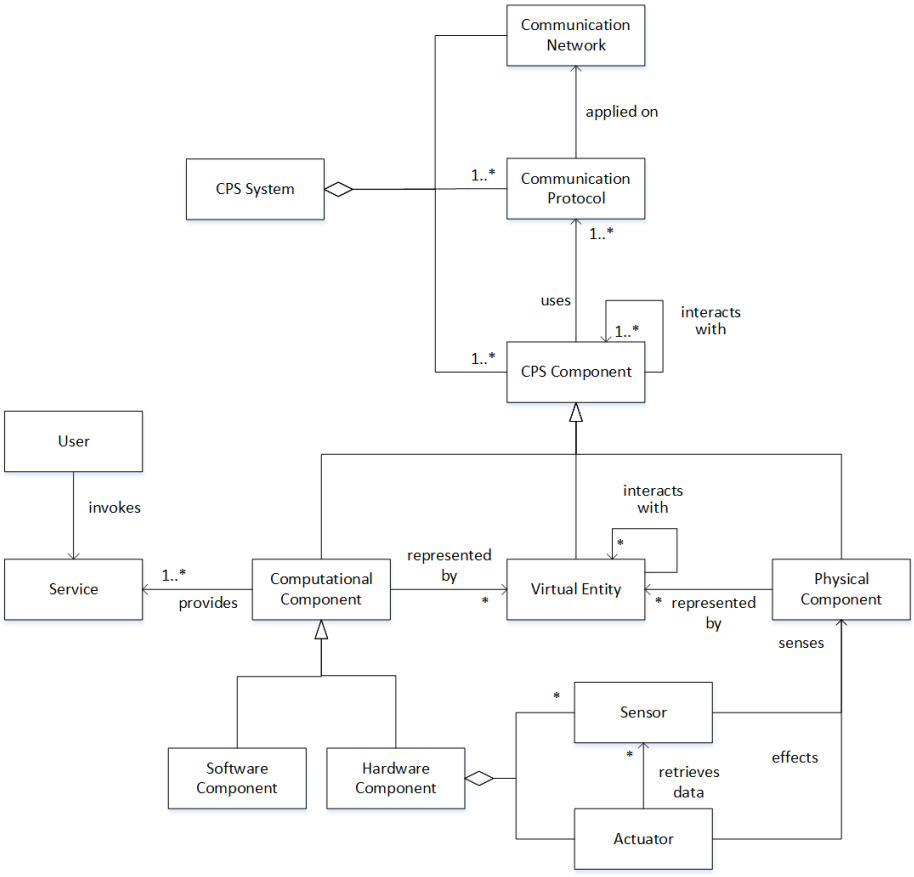

Figure 3.1: CPS Metamodel

3.3 Feature Model of Cyber-Physical Systems

To define the ontology for CPS we have carried out a domain analysis process using feature modeling. In the following sections we first describe the metamodel for CPS and then present the CPS feature diagram.

3.3.1 Top-Level Feature Diagram

Figure 3.2 shows the top-level feature diagram for CPS consisting of 5 mandatory root features Constituent Element, non-functional requirements, Application Domain, Disciplines, and Architecture. The numbers indicate the number of sub-features of the corresponding root features. In the following sub-section we describe each root feature in detail.

3.3.2 CPS Constituent Elements

Figure 3.3 shows the feature diagram of the constituent elements of CPS. Based on the feature diagram a CPS has the mandatory constituent elements of cyber element, physical element, control element, and network element. Further, human element can be an optional substituent element. Below, we describe each feature.

Cyber Element Communication and control between any human, biological, social system and any artificial device. It refers to systems where feedback is essential. Following are sub-features of the Cyber Element that further describe it:

- **Software**: Software is a collection of data or computer instructions that tell the computer how to work. It includes programs, libraries, and related non-executable data such as online documentation. Software is further divided into:
  - **Application Software**: Application software (app for short) is a program or group of programs designed for end users. Application software is built for specific tasks. Application software is dependent on system software.
  - **System Software**: System software is software designed to provide a platform for other software. Examples of system software include **operating systems** like macOS, GNU/Linux and Microsoft Windows, **system services** like applications for file system optimization, and local network D2D communication, and **system utilities** like file copy, calculator, and various desktop accessories.
  - **Embedded Software (Firmware)**: Embedded software is computer software, written to control machines or devices that are not typically thought of as computers, commonly known as embedded systems. It is typically specialized for the particular hardware that

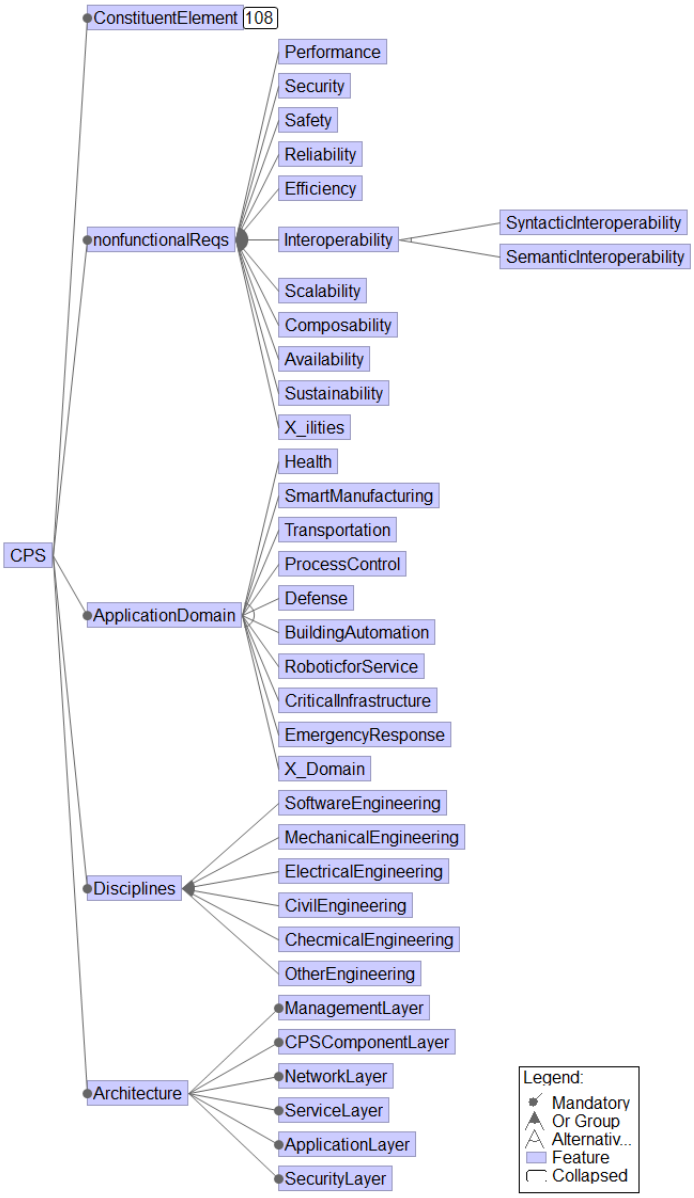

Figure 3.2: Top-Level Feature Diagram CPS

it runs on and has time and memory constraints. A precise and stable characteristic feature is that no or not all functions of embedded software are initiated/controlled via a human interface, but through machine-interfaces instead. Unlike standard computers that generally use an operating systems such as OS X, Windows or GNU/Linux, embedded software may use no operating system, or when they do use, a wide variety of operating systems can be chosen from, typically a real-time operating system.

Physical Element The physical entity that is controlled by the cyber element, consisting of the sensor(s), actuator(s), plant, controller(s), and environment. The controller can be of two kinds:

- **Mechanical Controller**: A mechanical control mechanism like a bimetallic strip in a thermostat. Its input is made up of mechanical effort.
- **Hardware Platform**: An electronic execution platform. It has the following components:
  - **Processor**: A processor is the electronic circuit which performs operations on some external data source, usually memory, or some other data stream. It typically takes the form of a microprocessor which is fabricated on a single MOS integrated circuit (IC). A processor can be a **Central Processing Unit (CPU)** if the design confirms to the Von Neumann architecture, and it contains atleast one control unit, arithmetic logic unit and processor registers. A processor can also be a **Graphics Processing Unit (GPU)** which is a specialized electronic curcuit designed to rapidly manipulate and alter memory to accelerate the creation of images intended for display output. A GPU can also be used for general purpose computing in which case it is called a 'GPGPU'.
  - **Memory**: Memory refers to a device that is used to store information for immediate use in a computer or related hardware device. It can refer to **Main Memory** which is the memory unit that communicates directly within the CPU. It is the central storage unit of the syste, is large and fast to store data during computer operations. It is made up of RAM and ROM. **Cache Memory** stores the data or contents of main memory that are repeatedly used by the CPU. Whenever the CPU needs to access memory, it first checks the cache memory and then the main memory. Devices that provide backup storage are called **Auxiliary Memory**. It is not directly accessible to the CPU and is accessed using I/O channels. magnetic tapes and disks are common auxiliary memory devices. Nowadays, important information is also backed up to remote storage systems, typically over the internet, in an arrangement called **cloud storage**.

- **System Bus**: The system bus is a pathway composed of cables and connectors used to carry data between a computer microprocessor and the main memory. The bus provides a communication path for the data and control signals moving between the major components of the computer system.
- **H/W Topology**: The way in which the constituents of the hardware platform are connected or interrelated. There could be multiple processing units, memory units, devices, sensors, and application-specific circuits that have to be connected to each other in a systematic manner. **External Interfaces** and connections in between components of the execution platform help define the arrangement.
- **Application-Specific Circuit**: (ASICs) Various application specific circuits like network adapters, chips designed to run digital voice recorders, high-efficiency bitcoin miners, etc.

Control Element The action of modifying the behavior of a system through feedback. The Control element is one of the most important and elaborate components of the CPS because it governs the interactions of the plant with its environment. Therefore, it becomes imperative to give a brief explanation of the Control element specifications i.e. its sub-features.

- **State**: The intrinsic configuration and description of the system.
- **Disturbance**: External influence of the environment in the system, typically unknown, but usually accounted for when designing the system.
- **Input**: Abstraction of the external factors in a system influencing its behavior.
- **Output**: Abstraction of the effect of a system on its environment.
- **Goal** The desired behavior of a system. It can either be modeled by a set-point, which is a static goal (that does not vary with time), or tracking the goal that varies with time. Another feature of the goal is its validity region - whether it is only valid in a subset of, or the complete state space of the system. The goal is reached by a regulation action. The goal is specified in implementations by the means of a reference signal, which models the desired value for the state of the system to reach.
- **Feedback**: The implementation of the control action by sensing the output/state of a system and modifying its input, by actuators, to meet a pre-defined control goal. The feedback is characterised by its dependency - whether the control action is designed depending on the system state or the system output. The scope of the feedback describes whether the feedback is applied to a centralised entity or not.

- **Dynamics**: This feature of the control element describes its evolution over time - its linearity, continuity, and time dependence (whether the system works in continuous or discrete time). Behavior that does not exist in single systems, but when many systems come together and interact, the systems form a complex system that results in an emergent behavior like swarming, flocking, collective, competition, etc. Such behavior also comes under the domain of control element dynamics. The topology i.e. the structure of interconnections of the components of the CPS can also be either static or adaptive.

- **Properties**: Characteristics of the CPS related specifically to the control element, like stability, passivity, robustness, adaptation, controllability, autonomy, intelligence, consistency, learning and uncertainty. Uncertainty can be deterministic, non-deterministic, probabilistic, or stochastic.

- **Diagnostics**: Ability to identify the properties of the CPS.

- **Prognostics**: The ability to predict a time in the future, when the CPS will not perform as expected, anymore.

Network Element A set of elements (for example, nodes) connected in some physical or abstract manner (for example, links). A network can adopt different configurations such as star, bush, ring, mesh, point to point, and hybrid. Further, a network may use different communication mechanisms including the communication type (synchronous or asynchronous), and communication protocol (P2P, Client-Server, Broker, other).

Human Element Humans within a CPS perform certain roles within the CPS. Each role represents some capacity or position, where humans playing the role need to contribute for achieving certain behavior goals set for the CPS. Each role is defined in terms of responsibilities and constraints pertaining to the role that are required for contributing to achieving the behavior goals set for the CPS. Responsibilities are components of a role that determine what a human performing the role must do for the behavior goals of the CPS to be achieved. Constraints are conditions that a human performing the role must take into consideration when exercising its responsibilities.

Humans within a CPS exercise their responsibilities defined by roles by performing certain actions. Action is an entity that is targeted at changing the state of the CPS or environment. Actions are divided into physical actions, communicative actions, and epistemic actions. Physical action is a kind of action that changes the state of a physical element of the CPS or environment. A communicative action is a kind of action that sends a message through a communication network of the CPS. An epistemic action is a kind of action that changes the state of the data held by the CPS. A human performs actions through actuators. A human can perceive events generated by the CPS or environment. An event is a kind of entity that is related to the states of affairs before

and after it has occurred. A human perceives events through sensors. State of affairs is a collective state of the entities of the CPS and the environment. An entity is anything perceivable or conceivable.

3.3.3 Non-Functional Requirements

Cyber-Physical Systems revolutionize our interaction with the physical world. Of course, this revolution does not come free. Since even legacy embedded systems require higher standards than general-purpose computing, we need to pay special attention to this next generation physically-aware engineered system requirements if we really want to put our full trust in them. Therefore, we want to clarify the definitions of some common CPS system-level requirements. In the top-level feature diagram of Figure 3.2 the branch non-functional requirements include all the identified relevant CPS non-functional requirements.

Accuracy Accuracy refers to the degree of closeness of a system's measured/observed out-come to its actual/calculated one. A highly accurate system should converge to the actual outcome as close as possible. High accuracy especially comes into play for CPS applications where even small imprecisions are likely to cause system failures. For example, a motion-based object tracking system under the presence of imperfect sensor conditions may take untimely control action based on incorrect object position estimation, which in return leads to the system failure.

Adaptability Adaptability refers to the capability of a system to change its state to survive by adjusting its own configuration in response to different circumstances in the environment. A highly adaptable system should be quickly adaptable to evolving needs/circumstances. Adaptability is one of the key features in the next generation air transportation systems (e.g. NextGen). NextGen's capabilities enhance airspace performance with its computerized air transportation network which enables air vehicles immediately to accommodate themselves to evolving operational environment such as weather conditions, air vehicle routing and other pertinent flight trajectory patterns over satellites, air traffic congestion, and issues related to security.

Availability Availability refers to the property of a system to be ready for access even when faults occur. A highly available system should isolate malfunctioning portion from itself and continue to operate without it. Malicious cyber-attacks (e.g. denial of service attacks) hinder availability of the system services significantly. For example, in Cyber-Physical Medical Systems, medical data shed light on necessary actions to be taken in a timely manner to save a patient's life. Malicious attacks or system/component failure may cause services providing such data to become unavailable, hence, posing risk on the patient's life.

Composability Composability refers to the property of several components to be merged within a system and their inter-relationships. A highly composable system should allow re-combination of the system components repeatedly to satisfy specific system requirements. Composability should be examined in different levels (e.g. device composability, code composability, service compos-

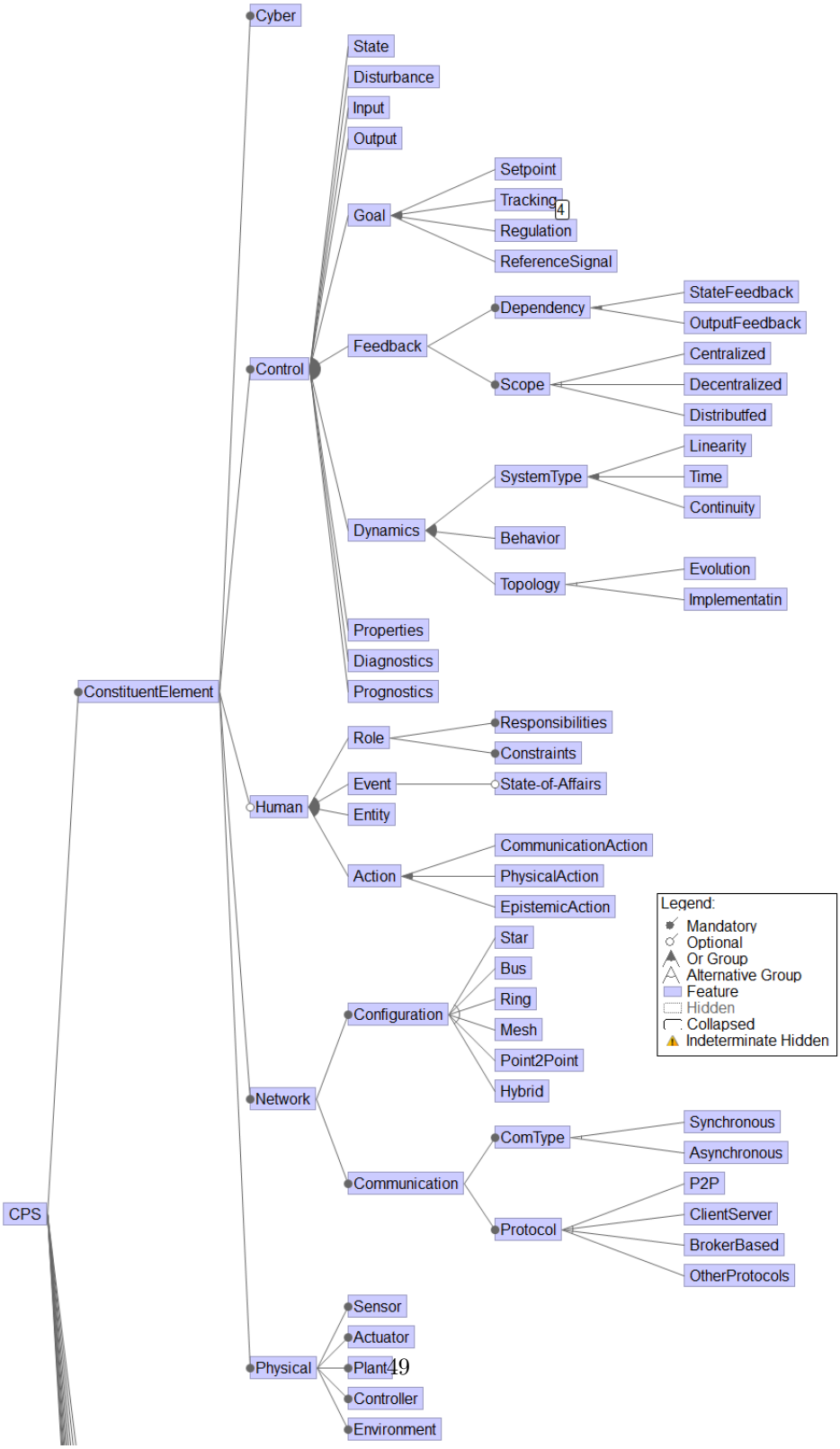

Figure 3.3: Feature Diagram CPS Constituent Elements

ability, system composability). Certainly, system composability is more challenging, hence the need for well-defined composition methodologies that follow composition properties from the bottom up. Additionally, requirements and evaluations must be composable accordingly. In the future, it will probably be of paramount importance to incrementally add emerging systems to the system of systems (e.g. CPS) with some predictable confidence with-out degrading the operation of the resulting system.

Compositionality Compositionality refers to the property of how well a system can be understood entirely by examining every part of it. A highly compositional system should provide great insight about the whole from derived behaviors of its constituent parts/components. Achieving high compositionality in CPS design is very challenging especially due to the chaotic behavior of constituent physical subsystems. Designing highly compositional CPS involves strong reasoning about the behavior of all constituent cyber and physical subsystems/components and devising cyber-physical methodologies for assembling CPSs from individual cyber and physical components, while requiring precise property taxonomies, formal metrics and standard test benches for their evaluation, and well-defined mathematical models of the overall system and its constituents.

Confidentiality Confidentiality refers to the property of allowing only the authorized parties to access sensitive information generated within the system. A highly confidential system should employ the most secure methods of protection from unauthorized access, disclosure, or tampering. Data confidentiality is an important issue that needs to be satisfied in most CPS applications. For example, in an emergency management sensor network, attacks targeting confidentiality of data transmitted may degrade effectiveness of an emergency management system. Confidentiality of data transmitted through attacked sensor nodes can be compromised and that can cause data flow in the network to be directed over compromised sensors; critical data to be eavesdropped; or fake node identities to be generated in the network. Further, false/malicious data can be injected into the network over those fake nodes. There-fore, confidentiality of data circulation needs to be retained in a reasonable degree.

Dependability Dependability refers to the property of a system to perform required functionalities during its operation without significant degradation in its performance and out-come. Dependability reflects the degree of trust put in the whole system. A highly dependable system should operate properly without intrusion, deliver requested services as specified and not fail during its operation. The words dependability and trustworthiness are often used interchangeably. Assuring dependability before actual system operation is a very difficult task to achieve. For example, timing uncertain-ties regarding sensor readings and prompt actuation may degrade dependability and lead to unanticipated consequences. Cyber and physical components of the system are inherently interdependent and those underlying components might be dynamically interconnected during system operation, which, in return, renders dependability analysis very difficult. A common language to express dependability related information across constituent systems/underlying components should be introduced in the

design stage.

Efficiency Efficiency refers to the amount of resources (such as energy, cost, time etc.) the system requires to deliver specified functionalities. A highly efficient system should operate properly under optimum amount of system resources. Efficiency is especially important for energy management in CPS applications. For example, smart buildings can detect the absence of occupants and turn off HVAC (Heating, Ventilation, and Air Conditioning) units to save energy. Further, they can provide automated pre-heating or pre-cooling services based on the occupancy prediction techniques.

Heterogeneity Heterogeneity refers to the property of a system to incorporate a set of different types of interacting and interconnected components forming a complex whole. CPSs are inherently heterogeneous due to constituent physical dynamics, computational elements, control logic, and deployment of diverse communication technologies. Therefore, CPSs necessitate heterogeneous composition of all system components. For example, incorporating heterogeneous computing and communication capabilities, future medical devices are likely to be interconnected in increasingly complex open systems with a plug-and-play fashion, which makes a heterogeneous control network and closed loop control of interconnected devices crucial. Configuration of such devices may be highly dynamic depending on patient-specific medical considerations. Enabled by the science and emerging technologies, medical systems of the future are expected to provide situation-aware component autonomy, cooperative coordination, real-time guarantee, and heterogeneous personalized configurations far more capable and complex than today's.

Integrity Integrity refers to the property of a system to protect itself or information within it from unauthorized manipulation or modification to preserve correctness of the information. A high integrity system should provide extensive authorization and consistency check mechanisms. High integrity is one of the important properties of a CPS. CPSs need to be developed with greater assurance by providing integrity check mechanisms on several occasions (such as data integrity of network packets, distinguishing malicious behaviors from the ambient noise, identifying false data injection and compromised sensor/actuator components etc.). Properties of the physical and cyber processes should be well-understood and thus can be utilized to define required integrity assurance.

Interoperability Interoperability refers to the ability of the systems/components to work together, exchange information and use this information to provide specified services [41]. A highly interoperable system should provide or accept services conducive to effective communication and interoperation among system components. Performing far-reaching battlefield operations and having more interconnected and potentially joint-service combat systems, Unmanned Air Vehicles (UAVs) call for seamless communication between each other and numerous ground vehicles in operation. The lack of interoperability standards often causes reduction in the effectiveness of complicated and critical missions. Likewise, according to changing needs, dynamic standards should be developed and tested for devices, systems, and processes used in the Smart Grid to en-

sure and certify the interoperability of those ones being considered for a specific Smart Grid deployment under realistic operating conditions.

Maintainability Maintainability refers to the property of a system to be repaired in case a failure occurs. A highly maintainable system should be repaired in a simple and rapid manner at the minimum expenses of supporting resources, and free from causing additional faults during the maintenance process. With the close interaction among the system components (e.g. sensors, actuators, cyber components, and physical components) underlying CPS infrastructure, autonomous predictive /corrective diagnostic mechanisms can be proposed. Continuous monitoring and testing of the infrastructure can be performed through those mechanisms. The outcome of monitoring and testing facilities help finding which units need to be repaired. Some components, which happen to be the source of recurrent failures, can be redesigned or discarded and replaced with the ones with better quality.

Predictability Predictability refers to the degree of foreseeing of a system's state/behavior/functionality either qualitatively or quantitatively. A highly predictable system should guarantee the specified outcome of the system's behavior/functionality to a great extent every moment of time at which it is operating while meeting all system requirements. In Cyber-Physical Medical Systems (CPMS), smart medical devices together with sophisticated control technologies are supposed to be well adapted to the patient's conditions, predict the patient's movements, and change their characteristics based on context awareness within the sur-rounding environment. Many medical devices perform operations in real-time, satisfying different timing constraints and showing diverse sensitivity to timing uncertainties (e.g. delays, jitters etc.). However, not all components of CPMS are time-predictable. Therefore, in addition to new programming and networking abstractions, new policies of resource allocation and scheduling should be developed to ensure predictable end-to-end timing constraints.

Reconfigurability Reconfigurability refers to the property of a system to change its configurations in case of failure or upon inner or outer requests. A highly reconfigurable system should be self-configurable, meaning able to fine-tune itself dynamically and coordinate the operation of its components at finer granularities. CPSs can be regarded as autonomously reconfigurable engineered systems. Remote monitoring and control mechanisms might be necessity in some CPS application scenarios such as international border monitoring, wildfire emergency management, gas pipeline monitoring etc. Operational needs (e.g. security threat level updates, regular code updates, efficient energy management etc.) may change for such scenarios, which calls for significant reconfiguration of sensor/actuator nodes being deployed or the entire net-work to provide the best possible service and use of resources.

Reliability Reliability refers to the degree of correctness which a system provides to perform its function. The certification of system capabilities about how to do things correctly does not mean that they are done correctly. So a highly reliable system makes sure that it does the things right. Considering the fact that CPSs are expected to operate reliably in open, evolving, and uncertain environments, uncertainty in the knowledge, attribute (e.g. timing),

or outcome of a process in the CPS infrastructure makes it necessary to quantify uncertainties during the CPS design stage. That uncertainty analysis will yield to effective CPS reliability characterization. Besides, accuracy of physical and cyber components, potential errors in design/control flow, cross-domain network connections in an ad-hoc manner limit the CPS reliability.

Resilience Resilience refers to the ability of a system to persevere in its operation and delivery of services in an acceptable quality in case the system is exposed to any inner or outer difficulties (e.g. sudden defect, malfunctioning components, rising workload etc.) that do not exceed its endurance limit. A highly resilient system should be self-healing and comprise early detection and fast recovery mechanisms against failures to continue to meet the demands for services. High resilience comes into play in delivering mission-critical services (e.g. automated brake control in vehicular CPS, air and oxygen flow control over an automated medical ventilator etc.). Mission critical CPS applications are often required to operate even in case of disruptions at any level of the system (e.g. hardware, software, network connections, or the underlying infrastructure). Therefore, designing highly resilient CPS requires thorough understanding of potential failures and disruptions, the resilience properties of the pertinent application, and system evolution due to the dynamically changing nature of the operational environment.

Robustness Robustness refers to the ability of a system to keep its stable configuration and withstand any failures. A highly robust system should continue to operate in the presence of any failures without fundamental changes to its original configuration and prevent those failures from hindering or stopping its operation. In addition to failures, the presence of disturbances possibly arising from sensor noises, actuator inaccuracies, faulty communication channels, potential hardware errors or software bugs may degrade overall robustness of CPS. Lack of modeling integrated system dynamics (e.g. actual ambient conditions in which CPSs operate), evolved operational environment, or unforeseen events are other particular non-negligible factors, which might be unavoidable in the run-time, hence the need for robust CPS design.

Safety Safety refers to the property of a system to not cause any harm, hazard or risk in-side or outside of it during its operation. A very safe system should comply with both general and application specific safety regulations to a great extent and deploy safety assurance mechanisms in case something went wrong. For example, among the goals for Smart Manufacturing (SM), pointing-time tracking of sustainable production and real-time management of processes throughout the factory yield to improved safety. Safety of manufacturing plants can be highly optimized through automated process control using embedded control systems and data collection frame-works (including sensors) across the manufacturing enterprise. Smart networked sensors could detect operational failures/anomalies and help prevention of catastrophic incidents due to those failures/anomalies.

Scalability Scalability refers to the ability of a system to keep functioning well even in case of change in its size/increased workload, and take full advantage of it. The increase in the system throughput should be proportional to the

increase in the system re-sources. A highly scalable system should provide scatter and gather mechanisms for workload balancing and effective communication protocols to improve the performance. Depending on their scale, CPSs may comprise over thousands of embedded computers, sensors, and actuators that must work together effectively. Scalable em-bedded many-core architectures with a programmable interconnect network can be deployed to deliver increasing compute demand in CPS. Further, a high performance and highly scalable infrastructure is needed to allow the entities of CPS to join and leave the existing network dynamically. In the presence of frequent data dissemination among those entities, dynamic software updates (i.e. changing the computer program in run-time) can help update CPS applications dynamically and use CPS resources more productively.

Security Security refers to the property of a system to control access to the system re-sources and protect sensitive information from unauthorized disclosures. A highly secure system should provide protection mechanisms against unauthorized modification of information and unauthorized withholding of resources, and must be free from disclosure of sensitive information to a great extent. CPSs are vulnerable to failures and attacks on both the physical and cyber sides, due to their scalability, complexity, and dynamic nature. Malicious attacks (e.g. eavesdropping, man-in-the-middle, denial-of-service, injecting fake sensor measurements or actuation requests etc.) can be directed to the cyber infrastructure (e.g. data management layer, communication infrastructure, decision making mechanisms etc.) or the physical components with the intent of disrupting the system in operation or stealing sensitive information. Making use of a large-scale network (such as the Internet), adopting insecure communication protocols, heavy use of legacy systems or rapid adoption of commercial off-the-shelf (COTS) technologies are other factors which make CPSs easily exposed to the security threats.

Sustainability Sustainability means being capable of enduring without compromising requirements of the system, while renewing the system's resources and using them efficiently. A highly sustainable system is a long lasting system which has self-healing and dynamic tuning capabilities under evolving circumstances. Sustainability from energy perspective is an important part of energy provision and management policies. For example, the Smart Grid facilitates energy distribution, management, and customization from the perspective of customers or service providers by incorporating green sources of energy extracted from the physical environment. However, intermittent energy supply and unknown/ill-defined load characterization hinders the efforts to maintain long-term operation of the Smart Grid. To maintain sustainability, the Smart Grid requires planning and operation under uncertainties, use of real-time performance measurements, dynamic optimization techniques for energy us-age, environment-aware duty cycling of computing units, and devising self-contained energy distribution facilities (such as autonomous micro grids).

3.3.4 Application Domains

In the top-level feature diagram of Figure 3.2 the branch Application Domain shows the important application domains for CPS. A CPS can be applied for various application domains including Health, Smart Manufacturing, Transportation, Process Control, Defense, Building Automation, Robotic Services, Critical Infrastructure, Emergence Response, etc. In principle the list is open ended. Any physical system that is integrated and controlled by a cyber part can be considered as CPS.

3.3.5 Disciplines

In the top-level feature diagram of Figure 3.2 the branch Disciplines shows the important disciplines for CPS. In essence, CPS requires a holistic systems engineering approach. Systems engineering on its turn is an interdisciplinary approach that focuses on how to design, integrate, and manage complex systems over their life cycles. With this, CPS is inherently related to multiple disciplines including software engineering, mechanical engineering, electrical engineering, civil engineering, chemical engineering and others.

3.4 Architecture of CPS

The architecture of a CPS represents the gross level structure of the system consisting of cyber-physical components. Current architecture design approaches for CPS seem to be primarily domain-specific and no standard reference architecture has been yet agreed upon. In this line, the development of an ontology for CPS also contributes to the efforts for designing a reference architecture.

A CPS reference architecture defines the generic structure of CPS architectures for particular application domains, laying the foundation for functionality, dependability, and other quality properties. An architecture organizes the functionality and the properties of a system to enable partitioning, verification, and management. Figure 15 presents a layered view of a CPS architecture inspired on the IoT stack that arranges a CPS into successive layers of cohesive modules that share similar concerns. The four layers at the center include device layer, network layer, CPS component layer, application layer, and business layer. The CPS component layer includes the capabilities for the CPS components to undertake sensing and actuation. The network layer provides functionality for net-working connectivity and transport capabilities enabling the coordination of components. The Services layer consists of functionality for generic support services (such as data processing or data storage), and specific support capabilities for the particular applications that may already apply a degree of intelligence. The application layer orchestrates the services to provide emergent properties. Then, there are two main cross-cutting concerns. A Security layer captures the security functionality, while the management layer supports capabilities such as device management, traffic and congestion management.

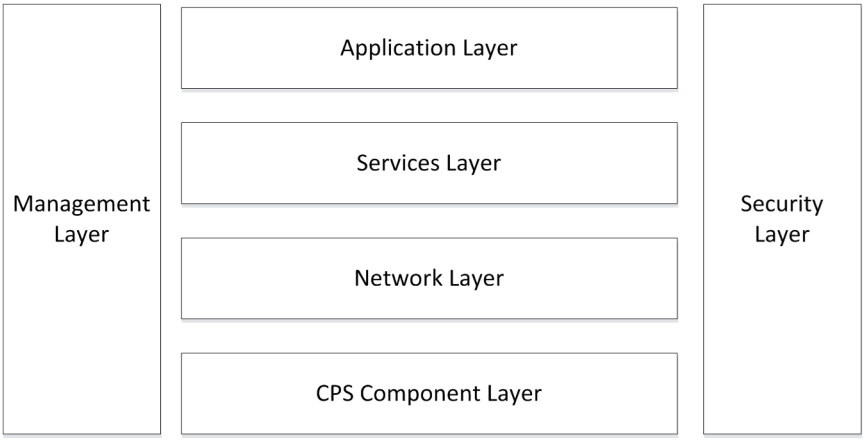

Figure 3.4: CPS Architecture

3.5 Examples

To demonstrate the application of the Cyber-Physical System concepts emulated in this chapter, the configuration of the CPS Feature Model (Section 3.3) pertaining to the examples described in Chapter 2 namely, Ensemble-Based Cyber-Physical System (Section 2.6.1) and HPI Cyber-Physical Systems Lab Autonomous Robot Case Study (Section 2.6.2) will be presented. A configuration is a particular instantiation of a Feature Model. It describes the features that are present or not present, in the particular product or system. Essentially, the Feature Model describes the inter-dependencies and constraints to be maintained in the features of each member of a family of systems and the configuration describes which features are present in the specific member. A configuration is deemed valid if it satisfies all the constraints of the parent Feature Model. The list of features, their inclusion or exclusion in the configuration, and the descriptions of the features pertaining to the example systems are given below:

3.5.1 Ensemble-Based Cyber-Physical System

In order to present all the related parts in the example presented in Chapter 2, we refer to the tasks related to vehicle joining a road train as (1), and vehicle finding a parking slot as (2).

- ✓**ConstituentElement** : (mandatory) The elements constituting the system
  - ✓**Cyber**: (mandatory) System model based on DEECo concepts.

    - ✓**Software**: Collection of data or computer instructions that tell the computer/controller of the robot, how to work.
        - ✓**Application Software**: The simulation software used to simulate traffic conditions and the EBCPS.
        - ✓**System Software**: The system software is the software running on the simulation computer and all of its included **System Services** and **System Utilities**.
        - ✘**Embedded Software (Firmware)**
- ✓**Control**: (mandatory) (1)(2) Vehicles have PID controllers to maintain the desired speed and the desired distance based on the driver preferences or the determined values in each vehicle mode in addition to (2) heading to the destination.
    - ✓**State**: We determine the states, which are the operational modes, of (1) a vehicle in a road train to be "Cooperative Adaptive Cruise Control (CACC)" or "Adaptive Cruise Control (ACC)", and of (2) a vehicle parks in a city to be "waiting", "search for parking lot", "reserve parking slot", "cancel reserved parking slot", "parking slot reserved", "search for parking slot on spot", "find parking slot on spot", "parking", "parked", or "leave the parking slot". The (2) parking lot has the following states for each parking slot: "available", "reserved", "canceled", "filled".
    - ✓**Disturbance**: (1)(2) Noise in sensors measurements, (1)(2) communication problems, and (1) traffic fluctuation.
    - ✓**Input**: (1) In road train, the vehicle receives the speed and the position of the vehicle in front. (2) Regarding finding a parking slot, the input is the availability of the parking lots, the available slots detected by other vehicles,
    - ✓**Output**: (2) In road train, the vehicle speed and distance from the vehicle in front is maintained. (2) When reaching the parking slot, the vehicle parks.
    - ✓**Goal**: (1) Hard real-time goal keep a save distance, (1) Soft real-time goal save fuel, and (2) Soft real-time goal is parking.
        - ✓**Setpoint**: (1) The desired speed and the desired distance from the vehicle in front. (2) The final destination and the parking stops in between for the vehicle.
        - ✓**Tracking**: The vehicle checks for the driver's inputs for (1) the desired speed and (1)(2) the desired stops, or (1) updates from the vehicles in the road train for the speed and distance to maintain. (2) The vehicle receives information about available parking slots from other vehicles or from parking lots. The **ValidityRegion** of the tracking in (1) the vehicle is local when vehicle is not in a road train and decentralized when vehicle is in a road train. Additionally, (2)

the vehicle tracks the available parking slots in decentralized manner. The **ValidityRegion** of the tracking in (2) the parking lot functions in decentralized manner.

- ✓**Regulation**: (1)(2) The vehicles communicate with each other to maintain the road train or to find an available parking slots on the spot. (2) The vehicle communicates with the parking lot to reserve a parking slot.
- ✓ **Reference Signal**: (1)(2) The GPS signal is a reference signal in vehicles for self-positioning and (2) calculating the distance from the stops and parking lots.

* ✓**Feedback**: (1) There are two kinds of feedback loops in the autonomous vehicle. The first is part of the control loop to regulate the speed and the safety distance. We represent that loop using MAPE-K model [42]. The second feedback learns about the vehicle behavior and the reliability of the sensors to perform better adaptation.
  - ✓**Dependency**: (1) The first feedback for the system is the measured acceleration from vehicle accelerometer (i.e. output of the Plant), which feeds back into the PID controller. The second feedback is studying the vehicle behavior and the reliability of the sensors to decide adapting to a more suitable mode for the current situation (e.g. change from CACC to ACC in case the wifi communication is unreliable).
  - ✓**Scope**: (1) The scope of the control signals is local to the vehicle.
* ✓**Dynamics** Context-based ensembles i.e. The vehicles can form in dynamic groups to (1) maintain distance in a road train or (2) detect available parking slots, and (2) the vehicles form a dynamic grouping with the parking lots to exchange information about the parking slots.
  - ✓**SystemType**: (**Linearity**) (1) The vehicle movement is non-linear. (**Time**) The system is discrete with relation to (1) the control-loop in the vehicle and the communication with (1)(2) other vehicles or (2) the parking lots.. (**Continuity**) The vehicle movement is continuous in reality. However, in the simulation it is discretized.
  - **Behaviour** (**Equilibria**) There exist multiple **Distributed** equilibria for each car. The **(Emergent Behavior)** of the CPS.
  - **Topology** (**Evolution**) (1)(2) The connections between the components are dynamic over time (i.e. through ensembles), which (**Implementation**) support adding constraints over the connections that provide context-awareness into the system.

    - ✓**Properties**: Autonomy, adaptation, learning and uncertainty. (**Uncertainty**) In the example, we consider white noise in measurements and network delays (i.e. stochastic and exponential distribution accordingly).
    - ✘ **Diagnostics**: No automatic diagnostic.
    - ✓**Prognostics**: The prediction is used in the system in the adaptation decision. For instance, after learning the vehicle behavior, (1) the vehicle decide to change mode to ACC because the WiFi communication is unreliable.
- ✓**Human**: The driver determine the next tasks to be performed in the vehicle.
    - ✓**Role**: (1) The driver decide to join or leave a road train, and (2) determine the final destination of the trip and the stops in between.
    - ✓**Event**: (1) Detecting a road train, (2) Determine the locations to visit.
    - **Entity**: The driver.
    - ✓**Action**: the driver can make a (1) Request for joining or leaving a road train, or a (2) Request for finding a parking slot.
- ✓**Network**: (mandatory) The communication is Peer-to-Peer communication (i.e. MANET-based wireless and IP-based communication)
    - ✓**Configuration**: (mandatory) The communication of peer-to-peer communication.
    - ✓**Communication**: (mandatory) The communication is governed by protocols and constraints.
        - ✓**ComType**: (mandatory) asynchronous communication since the communication is implicit, where the components propagate their knowledge and do not wait for an answer. However, we assume a shared clock for all the components in the example.
        - ✓**Protocol**: (mandatory) Gossiping via (2) UDP on top of Ethernet NIC and (1)(2) broadcast via wireless NIC.
- ✓**Physical**: (mandatory) The physical elements of the vehicle, the parking lots.
    - ✓**Sensor**: (mandatory) (1) When vehicle is on the road the used sensors are: wifi antenna, GPS antenna, camera, radar, lidar, ultrasonic, and accelerometer. (2) When the vehicle is looking for parking the camera in the vehicles, and (2) camera or sensors in the parking lots to detect the availability of a parking slot.
    - ✓**Actuator**: (mandatory) In the vehicle (1) the gas and brake pedals (i.e. vehicle engine). Even though the publications did

not cover the automatic parking , however it is interesting to highlight the (2) steering as actuators in the vehicle.

        * ✓**Plant**: (mandatory) (1) The vehicle movement equations (because we simulate)
        * ✓**Controller**: (mandatory) (1) The PID controllers
            - ✘**Mechanical**
            - ✓**Hardware Platform**: Consists of **Processor**, **Memory**, **System Bus**, **Hardware Topology**, and **Application-specific Circuit**
        * ✓**Environment**: (mandatory) (2) The cities, and (1)(2) the roads.

- ✓**nonfunctionalReqs** : Safety, Efficiency, Adaptability
- ✓**ApplicationDomain** : Transportation
- ✓**Disciplines** : The disciplines associated with the vehicle are - **Software Engineering** and **Mechanical Engineering**

3.5.2 HPI Cyber-Physical Systems Lab

- ✓**ConstituentElement** : (mandatory) The elements constituting the system
    - ✓**Cyber**: (mandatory) The cyber element of the system in the RTAI Linux OS.
        * ✓**Software**: Collection of data or computer instructions that tell the computer/controller of the robot, how to work.
            - ✓**Application Software**: In different stages, different application software are used to simulate and prototype the robot. These application software include FESTO Robotino Sim, Simulink, dSPACE System Desk etc.
            - ✓**System Software**: The system software is the software running on the robot controller which is RTAI Linux OS, and all of its included **System Services** and **System Utilities**.
            - ✓**Embedded Software (Firmware)**: The C code generated by dSPACE System Desk that is executed on the robot in the Software-in-the-Loop, and Hardware-in-the-Loop stages of development.
    - ✓**Control**: (mandatory) The robot has a control system that moderates the physical elements according to the different feedback inputs.
        * ✓**State**: The robots current position and orientation, and the task which it is performing (moving around, transporting pucks, or charging batteries) defines its state.

* ✓**Disturbance**: Obstacle while performing task, or the reception of a new task which disturbs the current state of the robot (like battery level falling below a specified level)
* ✓**Input**: Task to be performed.
* ✓**Output**: Task performed (movement of pucks).
* ✓**Goal**: Soft real-time goals are movement of pucks, ensuring battery is not depleted of charge. Hard real-time goal is avoidance of obstacles.
  - ✓**Setpoint**: The co-ordinates for the robot to reach.
  - ✓**Tracking**: The robot checks for new tasks or co-ordinates that may have been updated by the administrator. The **ValidityRegion** of the tracking is **local** to the robot. It only tracks for instructions concerning itself.
  - ✓**Regulation**: The actuators are activated in a co-ordinated manner to achieve the goals.
  - ✘ **Reference Signal**: There is no reference signal because the robot doesn't know if it's movement of the pucks is as desired i.e. succesful.
* ✓**Feedback**: There are various feedbacks for the control system.
  - ✓**Dependency**: There are both, **StateFeedbacks** i.e. the feedbacks that describe the state of the robot, for example, the co-ordinates and orientation, the battery level and **OutputFeedbacks** i.e. describe the output of the robot like the feedback from the incremental encoder of the actuators that give information about the actuator's executed movements.
  - ✓**Scope**: The scope of all the feedback signals is **centralized** to the control unit.
* ✓**Dynamics** How the system behaves over time.
  - ✓**SystemType**: (**Linearity**) The control-system uses non-linear signal processing equations. (**Time**) The system is discrete because it functions w.r.t edges of the clock. (**Continuity**) The control system is multi-modal continuous, but there is no direct interaction between the various dynamic systems of the robot (for eg. the omni-wheel drive, gripper, sensors, etc.) which is typical of such systems.
  - ✓**Behaviour**: (**Equilibria**) There exists a **Common** equilibrium between all the different dynamic components, governed by the central control system. The (**Emergent Behaviour**) of the robot system is **Cooperation** as it ultimately cooperates with the various other robots and entities that may be present in the environment.
  - ✓**Topology**: (**Evolution**) The interconnections between the different components of the robot stay the same over

time, they are **Static**. (**Implementation**) The connections between the components are **Physical**.
    - ✓**Properties**: **Stability**, **Dissipativity** (energy in battery is dissipated over time), **Robustness** (object detection and path-planning), **Controllability**, **Observability**, **Resilience**, **Autonomy**, **Consistency**.
    - ✘ **Diagnostics**: No automated diagnostic capability is present in the robot control system.
    - ✓**Prognostics**: Prognostics exist. That is why the indoor GPS-like architecture has been used so that the robot functions correctly according to the goal.
- ✓**Human**: The human operator defines the tasks for the robot.
    - ✓**Role**: The administrator has the **Responsibility** of providing the co-ordinates for the robot to work with. The administrator does so with specific **Constraints** like reachability, correctness and syntax.
    - ✓**Event**: Decision of the co-ordinates for the robot to pick-up and drop pucks.
    - ✓**Entity**: The administrator.
    - ✓**Action**: The human administrator performs a **Communication Action** where he/she communicates the required co-ordinates for the robot through a communication network.
- ✓**Network**: (mandatory) The network used in the system for different components to communicate with each other.
    - ✓**Configuration**: (mandatory) The robot network has a star-type topology. All the components are connected to the central processing unit.
    - ✓**Communication**: (mandatory) The communication is governed by protocols and constraints.
        - ✓**ComType**: (mandatory) The communication is **Synchronous** since the sensors, actuators and the processing unit share the same clock.
        - ✓**Protocol**: (mandatory) The network has some **Other Protocols** - half-duplex to interact with the various infrared distance sensors and laser scanners, whereas a full duplex protocol to interact with the administrator, and the different actuators.
- ✓**Physical**: (mandatory) The physical elements of the robot.
    - ✓**Sensor**: (mandatory) Various sensors which provide feedback to the controller, like the infrared distance sensors, laser scanners, the incremental encoder from the drive unit of the actuators, the sensors comprising the indoor GPS-like navigation system, the communication antennae for communicating with the administrator, etc.

    - ✓**Actuator**: (mandatory) The servo motors, omni-directional drive system, gripper, etc.
    - ✓**Plant**: (mandatory) The robot.
    - ✓**Controller**: (mandatory)
        - ✘**Mechanical**
        - ✓**Hardware Platform**: Consists of **Processor**, **Memory**, **System Bus**, **Hardware Topology**, and **Application-specific Circuit**
    - ✓**Environment**: (mandatory) The HPI CPSLab environment where the robot operates.

- ✓**nonfunctionalReqs** : **Performance**, **Security**, **Safety**, **Reliability**, **Efficiency**, **Scalability**, **Composability**, **Availability**, **Sustainability**, **Others**
- ✓**ApplicationDomain** : Domain of application of the robot can be broadly put under **RoboticforService** i.e. Service Robot.
- ✓**Disciplines** : The disciplines associated with the robot are - **Software Engineering**, **Mechanical Engineering**, **Electrical Engineering**

3.6 Conclusion

In this chapter we have provided a feature-based ontology of cyber-physical systems. We have adopted feature modeling to represent the common and variant features of a CPS. The CPS feature model has been developed after a thorough domain analysis on CPS. Each feature branch and feature leaf has been carefully checked and described. The resulting feature model shows the configuration space for developing CPSs. We have used two different case studies on CPS and illustrated how to derive a concrete CPS configuration. Both case studies were applied after the feature model was designed but we were able to model all the features of the case studies, and did not need to adapt the CPS feature diagram. This was important to validate the external validity of the feature diagram. In our future work we will apply the CPS for other case studies as well.

The feature model is worthwhile for both researchers and practitioners. Researchers can identify the features of current CPSs and aim to identify novel features to enhance the CPS domain. The feature model can thus be used as a means to pave the way for further research in CPS. Practitioners can benefit from the resulting CPS by using it to understand and analyze existing systems and or develop novel CPSs. The CPS feature diagram has been derived based on a solid domain analysis. Further it has been validated by real world examples including Ensemble-Based Cyber-Physical System and HPI Cyber-Physical Systems Lab Autonomous Robot Case Study. As such, we can expect that the feature diagram is quite stable. However, in case of new development the feature diagram is adaptable and extensible to describe novel features.