Article

An Ontology-Based Framework for Cyber Range Representation

Vyron Kampourakis *, Michail Takaronis, Vasileios Gkioulos, Sokratis Katsikas

Posted Date: 28 February 2026

doi: 10.20944/preprints202602.2048.v1

Keywords: cyber range; ontology; framework; protégé; RDF/XML; SPARQL

 doi:10.20944/preprints202602.2048.v1

Article

An Ontology-Based Framework for Cyber Range Representation

Vyron Kampourakis *, Michail Takaronis, Vasileios Gkioulos and Sokratis Katsikas

Norwegian University of Science and Technology, 2802 Gjøvik, Norway
* Correspondence: vyron.kampourakis@ntnu.no

Abstract

Cyber Ranges (CRs) are complex socio-technical ecosystems, combining infrastructure resources, software services, learning mechanisms, and human-in-the-loop processes for cybersecurity training, education, and experimentation. However, their design and representation are conventionally described by diverse architectural representations and a lack of standardization, making them difficult to compare, integrate, and reason in an automated manner. This paper proposes a novel framework that uniquely integrates the structural, functional, informational, and decisional aspects of CR platforms, formalizing them into a common semantic framework. It models the architectural and learning characteristics of CRs, allowing the representation of design choices, operational processes, information resources, and capability development. The ontology is implemented using OWL 2 DL, which includes logical constraints and enables consistency checking and automated reasoning. Validation through instantiation and competency question assessment shows that the model allows for structured querying, traceability across abstraction levels, and capability-level reasoning. The findings indicate that ontology-based modeling can serve as a basis for more formalized CR configuration analysis and capability-focused evaluation of diverse CR platforms.

Keywords: cyber range; ontology; framework; protégé; RDF/XML; SPARQL

1. Introduction

The rapid digitalization that fuels the fourth industrial revolution and the emerging era beyond has transformed manufacturing, business processes, and Critical Infrastructures (CI) [1,2]. Apparently, this technological advance has unlocked new operational capabilities, but, on the downside, it has expanded the attack surface to a wide spectrum of threat actors. The high-profile cyberattacks, ranging from the famous Stuxnet [3] attack on the uranium centrifuges in Iran's Natanz facility to the Industroyer [4] and Industroyer2 [5] attacks on power grid infrastructure, have clearly shown that cyber threats are no longer limited to the traditional Information Technology (IT) domain. At the same time, purely IT-oriented yet highly sophisticated attacks have also highlighted the issue of systemic risk for digitally interconnected environments. Notably, the SolarWinds supply-chain compromise [6,7] enabled adversaries to infiltrate thousands of organizations worldwide, including government agencies and major enterprises. Similarly, the Colonial Pipeline ransomware attack [8] highlighted how a cyber attack on purely IT-oriented environments can have a downstream effect on physical operations on a national scale. Collectively, these incidents highlight the evolving threat landscape across both cyber-physical and purely IT-based infrastructures. Thus, the need to enhance cyber preparedness for the broad range of stakeholders has never been more pressing.

Yet in practice, executing cybersecurity assessments on actual systems, particularly those part of CIs, is prohibitively difficult and often inadmissible, considering that it could jeopardize system stability, continuity, and safety. In this direction, Cyber Ranges (CRs), as an evolution of traditional security testbeds, have emerged as a comprehensive and versatile solution that caters to a diverse audience, offering a multifaceted perspective on enhancing the overall cybersecurity posture [9,10].

 doi:10.20944/preprints202602.2048.v1

Overall, CRs offer various capabilities, including but not limited to security research, competence building, and education. In simple terms, they can be utilized to raise awareness and educate personnel within an organization about cybersecurity best practices, serving also as valuable platforms for identifying vulnerabilities and devising suitable security controls to safeguard the under-scrutinized system.

Despite the merits of CRs, there is still no definitive agreement about their design, development, or evaluation in relation to existing guiding principles and standards. Several initiatives have been attempted toward developing architectural blueprints and reference models [9,11–13] and methodologies for assessing both existing platforms and new and emerging options [9,14]; however, these contributions tend to be disjointed and heterogeneous. For instance, among the many areas of CR research published to date, only one has utilized an ontology-based approach [15]. Yet, this work focused only on defining the structure and flow of exercise scenarios related to the field of cybersecurity and did not address questions pertaining to the overall architecture, operations, or organization of the entire CR ecosystem. Therefore, there is considerable opportunity for future research on the development of a unified ontology-based framework that encompasses the entire CR ecosystem and utilizes a formalized, semantically enriched, and machine-readable framework for synthesizing all of the diverse elements of the CR ecosystem and enabling both tools and human experts to interpret, relate, validate, and compare them systematically and transparently.

Contribution: To fill this gap, this paper presents an ontology-based framework that organizes and semantically connects and relates the diverse layers, modules, functionalities, and roles involved in a CR ecosystem. Importantly, the proposed ontology does not constitute a direct representation or encoding of an existing architecture blueprint or taxonomy. Rather, it constitutes a semantic meta-model over architectural knowledge. In other words, structured classifications, taxonomies, or architectures of CRs are inherently semi-formal and lack logical semantics. The ontology introduced in this work aims to abstract and formalize such architectural knowledge into a machine-interpretable framework enriched with constraints, dependency relations, and inference mechanisms. Consequently, the contribution lies not in reproducing architectural diagrams in OWL, but in transforming architectural knowledge into a logically grounded model that supports automated reasoning, validation, and capability inference. The contribution of this work is structured in three phases as follows.

- **Phase1 (Modeling):** We develop the conceptual model of the core elements that constitute a CR. This includes translating widely used classifications and design aspects into a unified and computationally interpretable ontology that defines a common language for describing CRs in a consistent and unambiguous manner.
- **Phase2 (Implementation):** We implement the proposed conceptual model using Protégé [16] and formalize it in Resource Description Framework (RDF) and Extensible Markup Language (XML) formats, thereby producing a structured ontology that captures the relationships, dependencies, and compositional structures among the various design parameters of CRs. This implementation supports structured analysis, enhances transparency, and enables more informed and explainable decision-making during the CR design and evaluation process.
- **Phase3 (Validation):** We validate the proposed ontology through an indicative CR [17], demonstrating how the multi-layer modeling approach, which distinguishes between descriptive, operational, and capability-related aspects, provides a holistic and interpretable representation of CR ecosystems. This validation highlights the ontology's ability to facilitate comparability and repeatability across heterogeneous CR platforms and use cases.

The rest of the paper is structured as follows. The next section reviews the relevant literature in this field. Section 3 presents the necessary background on CRs and ontology engineering, establishing the conceptual foundations of this work. Section 4 introduces the proposed CR ontology model, detailing its modeling scope, class hierarchy, properties, and logical axioms. Section 5 describes the formal implementation of the model in OWL 2 Description Logic (DL) using Protégé and discusses the technical realization of its reasoning mechanisms. Section 6 validates the ontology through the

 doi:10.20944/preprints202602.2048.v1

instantiation of a CR platform and the evaluation of representative competency questions via SPARQL queries. Section 7 discusses the limitations of the proposed approach and outlines directions for future work. The last section concludes the paper.

2. Related Work

A substantial body of literature focuses on improving the automation, deployment, and scenario engineering capabilities of CRs. However, existing contributions address specific operational modules of CR ecosystems, emphasizing scenario development, infrastructure orchestration, and content automation.

The work in [18] introduced *Nautilus*, a tool that automates the packaging and deployment of CR exercises. Specifically, the authors provided a declarative description language to specify network topologies, services, vulnerabilities, and scoring logic to improve portability and reproducibility of exercises. Their contribution remains scenario-centric, namely, streamlining deployment workflows, and does not extend toward a formal representation of CR architectural elements or ecosystem-wide relationships. Similarly, in [19] the authors proposed the Cyber Range Instantiation System (CyRIS), allowing instructors to define CR configurations using high-level YAML specifications. Their approach automated virtual machine provisioning, network setup, and security content deployment. Although CyRIS aims to enhance repeatability and reduce manual effort, its orientation remains procedural and deployment-driven, rather than grounded in formal knowledge representation or ontological modeling.

Automation-oriented approaches were further explored in [20], where Infrastructure-as-Code (IaC) techniques were combined with CR platforms to enable reproducible vulnerability injection aligned with OWASP Top 10 categories [21]. Likewise, the authors in [22] proposed a Scenario Definition Language (SDL) built upon TOSCA [23] to declaratively model systems, vulnerabilities, and user roles and to enable automated deployment and runtime verification. Their efforts advanced scenario authoring and infrastructure orchestration, but they remain confined to implementation-level automation. The work in [24] surveyed automation practices in modern CR platforms, identifying common capabilities such as infrastructure provisioning, orchestration, adversary emulation, and performance scoring. While this study highlighted automation as a key enabler of scalability and fidelity, it did not introduce any kind of formal or machine-interpretable model for CR design or representation.

Model-driven and language-based automation has also been explored in [25], with the authors proposing Domain-Specific Language (DSL)-based scenario descriptions combined with logical validation and orchestration components. More recently, the authors in [26] leveraged Large Language Models (LLMs) enhanced with Retrieval-Augmented Generation (RAG) to automate scenario creation. Although these works aim to enhance automation and content generation, they focus solely on operational workflows rather than structured CR engineering models. Among the surveyed literature, the work in [15] represented a step toward formalization by proposing an OWL/RDF-based ontology for cybersecurity exercise scenarios. The proposed model supports semantic representation and SPARQL-based querying of scenario knowledge. However, its scope is limited to scenario modeling without extending to the broader CR ecosystem, including architectural layering, infrastructure orchestration, lifecycle processes, cross-layer dependencies, or capability modeling.

Table 1. Comparative overview of representative works on Cyber Range design and automation.

Ref.	Year	Focus	Scenario Modeling	Infr. Orch.	Knowledge Model	Coverage
[18]	2021	Exercise packaging & deployment	Y	Y	N	Partial
[19]	2016	CR instantiation automation	Y	Y	N	Partial
[20]	2025	IaC-based scenario deployment	Y	Y	N	Partial
[22]	2020	Declarative scenario specification (SDL)	Y	Y	N	Partial
[24]	2021	Automation survey & orchestration	N	Y	N	Limited
[25]	2022	Model-driven scenario automation	Y	Y	N	Partial
[26]	2024	LLM-assisted scenario generation	Y	N	N	Limited
[15]	2021	Scenario ontology modeling	Y	N	Y	Scenario-only
This Work	2026	Ontology-based CR engineering	Y	Y	**Y**	**End-to-end**

 doi:10.20944/preprints202602.2048.v1

As summarized in Table 1, existing research primarily concentrates on automation mechanisms, infrastructure orchestration, and scenario execution. Although these approaches aim to enhance usability, repeatability, and operational scalability, they remain implementation-driven and functionally scoped. As already mentioned in section 1, to the best of our knowledge, no work introduces a unified, semantically structured representation capable of modeling the CR ecosystem in its entirety. Therefore, this gap motivates the ontology-based approach proposed in this work.

3. Background

This section provides a brief, but concise overview of CR and ontologies, serving as a prerequisite and facilitating the subtle comprehension of the subsequent discussion and analysis in Sections 4 and 5.

3.1. Cyber Ranges

To ground this discussion, we adopt the work in [9] as a cornerstone. Their study proposed a reference architecture for CR design and development, consolidating state-of-the-art standards and guidelines, classifying core CR components across structural, functional, and informational layers, and introducing a preliminary evaluation formula for assessing architectural conformance. Importantly, as a highly relevant input to this study, the work in [9] consolidated different CR aspects through a quad-level classification, condensing the functional and informational aspects of common CR platforms in a structured manner. Moreover, the same classification incorporated decisional aspects, essential for the CR designer's preliminary architectural decisions before committing to any engineering or implementation actions. Generally, the classification of Figure 1, as adapted from [9], serves as an invaluable input for the construction of the ontology model later introduced in section 4.

Observe from figure 1 that each of the four horizontal dimensions outlines the CR aspects based on the different categories, that is, broad aspects, core infrastructure, Range Learning and Management System (RLMS), and user-oriented aspects. From a vertical standpoint, the same aspects are further classified into decisional, functional, and informational. Specifically, decisional aspects relate to the decision-making processes and strategic considerations important for the CR design phase, establishing a solid basis for making informed engineering and implementation choices. As depicted in the figure, they are further divided into infrastructure- and scope-decisional. The first class of decisional aspects includes the domain of application, type of infrastructure, virtualization technology, delivery model, scope, and governance; simply put, all decisions purely related to the infrastructure of the CR. The second class of decisional aspects regards the methods, evaluation practices, curricula type, and target didactics, and finally, defines user types and teams. Together, both infrastructure- and scope-decisional aspects are carefully considered by the CR designer to establish the purpose of the CR and to determine how it should ultimately be instantiated. Subsequently, depending on the infrastructure and scope decisions, the CR designer focuses on the functional and informational aspects of the CR. Namely, the core infrastructure's functional and informational aspects are resolved consistently with the infrastructure decisions. Accordingly, the RLMS functional and informational aspects are decided based on the infrastructure decisions and the scope decisions pertaining to the purpose of the CR, as well as the learning methods, evaluation practices, curricula type, and target didactics. Last, the user-oriented functional and informational aspects are determined in conformity with the infrastructure decisions and the scope decisions per the user types and teams.

3.2. Ontologies

Regarding ontologies, they can be defined as " [..] explicit formal specification of a conceptualization" [27]. In detail, an ontology formally represents and describes a given concept or system, along with all the relevant relationships among the entities that constitute the system of interest, aiming to disentangle and abstract it in a machine-interpretable manner [28]. Additionally, ontologies facilitate the exchange of semantic information between the system of interest and Humans-in-the-Loop (HitL), providing a common language and outlining all the concepts and inter-concept relations in a formal

 doi:10.20944/preprints202602.2048.v1

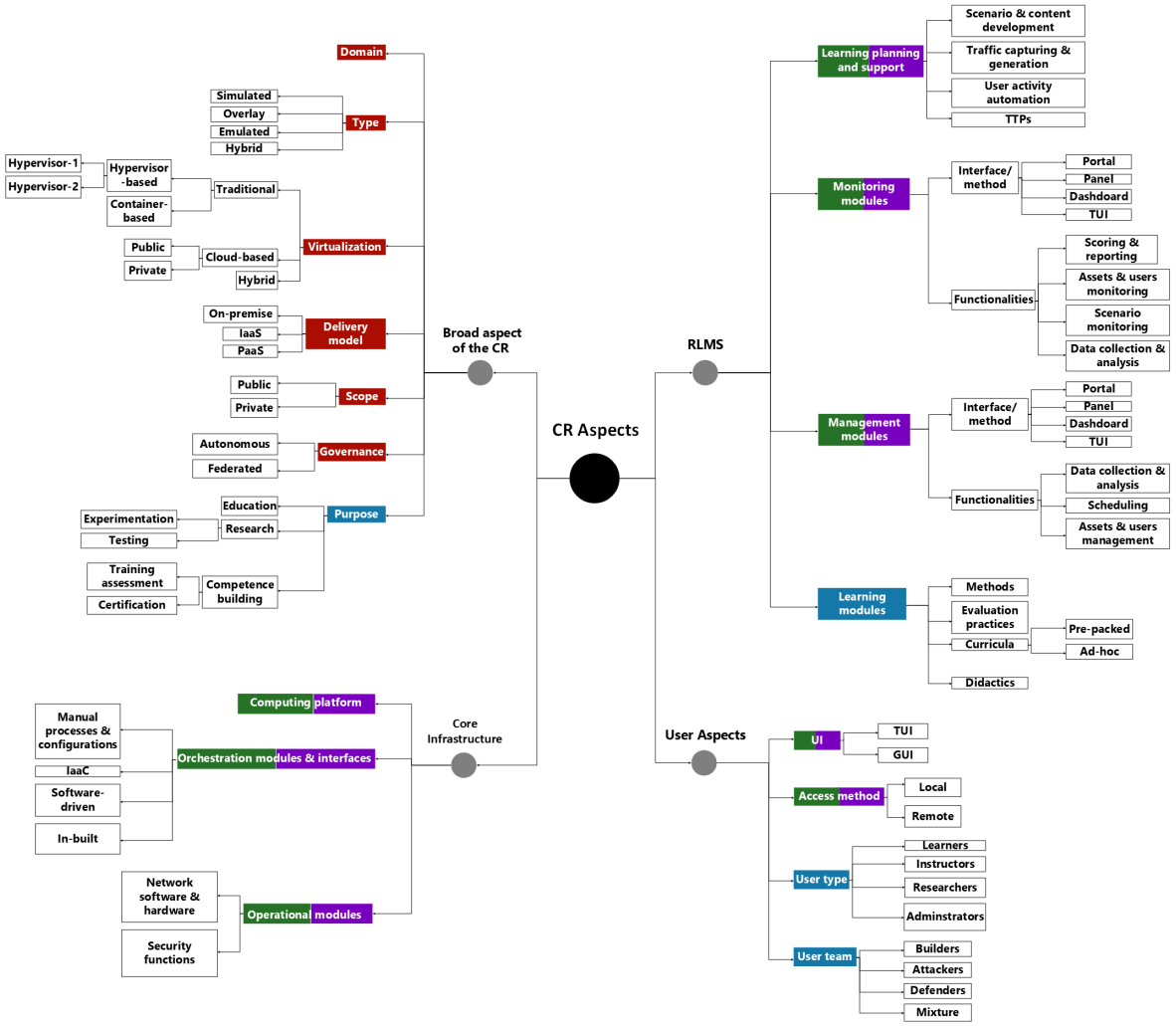


Figure 1. A quad-level classification of the CR aspects (adapted from [9]). With brown are the infrastructure-decisional, with blue the scope-decisional, with green the functional, and with purple the informational aspects.

logic representation. In a nutshell, ontologies can be expressed as graphs, whose vertices represent the concepts of interest and edges denote the relationships among them. Typically, such an ontology graph is structured hierarchically, similar to classes in object-oriented programming or structured tables in relational databases. Altogether, this formal representation allows the automated or machine-aided extraction and aggregation of knowledge from different sources and possibly in different formats [27].

Importantly, ontologies are fundamental entities for achieving interoperability among diverse organizations and stakeholders within – but not exclusively – the Semantic Web, as they are designed to formally capture domain knowledge. Simply put, their function is to make direct and implicit semantics clear in a generic and shared manner, thus providing a common basis for agreement and communication within a specific domain. To this end, Semantic Web technologies provide formal representation languages, such as RDF and the Web Ontology Language (OWL), enabling the description of the entity of interest through well-defined concepts and relationships rather than textual descriptions. Consequently, ontologies can be applied to a wide range of domains, including scientific knowledge portals, information management and integration systems, electronic commerce, and web services. Within a certain domain, they are primarily used for communication and knowledge sharing, logical inference and reasoning, and knowledge base development. Based on existing literature [29], we identify and summarize the benefits of developing and using ontologies in knowledge modeling as follows.

 doi:10.20944/preprints202602.2048.v1

- Provide a shared and common understanding of structured information among HitL and machines or software agents.
- Support the reuse of domain knowledge across applications and systems.
- Enable interoperability among heterogeneous models and domain-specific vocabularies.
- Facilitate and simplify communication among HitL, computational systems, and between them.
- Offer expressive capabilities for capturing contextual knowledge from diverse and heterogeneous sources.

Specifically in the realm of CRs, ontologies can play a key role in addressing their inherent complexity, heterogeneity, and multidimensionality. As discussed in section 3.1, CRs integrate diverse infrastructural components, functional modules, pedagogical mechanisms, operational workflows, and stakeholder roles. To this end, an ontology-based approach can enable the formalization of these elements into a unified, semantically enriched representation, making their interdependencies, constraints, and design rationale explainable and machine-interpretable to a large extent. Subsequently, supporting systematic reasoning over CR ecosystems could facilitate the comparison and validation of different CR implementations, possibly in tandem with existing evaluation approaches [14], thereby enhancing transparency during design and evaluation decision-making. Ultimately, ontology-based representation can serve as a cornerstone for interoperability among heterogeneous CR platforms, enabling knowledge reuse, automated analysis, and tool-assisted support for CR composition, capability assessment, and lifecycle management.

4. Ontology Model

This section introduces the conceptual foundation of the proposed CR ontology model. It first defines the modeling scope and abstraction boundaries of the ontology, and then presents the detailed model design, including its core classes, properties (relationships), and logical constraints that structure CR ecosystems in a semantically coherent manner.

4.1. Modeling Scope

The purpose of the ontology model that this section introduces is to provide a formal, semantically rich representation of CR ecosystems, capturing both their structural composition and the rationale behind their design. As discussed in Sections 1 and 3, CRs are inherently complex socio-technical systems, combining heterogeneous infrastructure components, software services, learning mechanisms, operational workflows, and human roles. Existing CR frameworks and reference architectures describe these elements largely in a semi-formal manner, limiting machine interpretability, comparison, and automated reasoning.

Namely, although the quad-level classification of figure 1 can serve as a value domain input, the ontology does not merely mirror this classification. Instead, it introduces an additional semantic layer that explicitly models dependencies, constraints, and cross-layer interactions that are not formally captured in the original taxonomic representation. Simply put, the ontology serves as a semantic abstraction layer that governs how architectural elements relate, constrain one another, and collectively enable higher-level capabilities.

To address these limitations, the ontology model is designed to serve as a conceptual bridge between high-level CR design decisions and their engineering and implementation, remaining generic to a certain degree to be able to accommodate diverse CR implementations and application domains. In simple terms, the ontology does not aim to prescribe a specific CR instance; instead, it establishes a common conceptual and semantic foundation upon which different CR configurations can be consistently described, analyzed, and compared. In this regard, based on the vertical classification of CR aspects, as shown figure 1, the ontology modeling scope can be defined as follows.

- ***Decisional aspects:*** They represent strategic, organizational, and architectural choices made at early CR design stages. In this sense, they define the purpose, scope, and constraints of the CR, including decisions related to infrastructure, including the delivery model, governance, and

 doi:10.20944/preprints202602.2048.v1

virtualization technologies, as well as decisions regarding the CR scope, such as target audience, pedagogical objectives, and evaluation practices. Collectively, decisional aspects condition and influence all subsequent functional and informational elements of the CR.

- ***Functional aspects:*** They capture the operational behaviors, services, and mechanisms provided by the CR to realize its intended objectives, as defined by the decisional aspects. Specifically, this class of aspects describes what the CR does at runtime, including functionalities such as simulation and emulation, orchestration and automation, monitoring, learning management, and user interaction.
- ***Informational aspects:*** They describe the data-centric dimension of the CR, encompassing the information objects that are generated, consumed, processed, or stored during its operation. Typical informational elements include scenarios, configuration artifacts, logs, telemetry data, learning content, and evaluation results. Keep in mind that informational aspects are explicitly linked to the functional aspects that produce or consume them.
- ***Capabilities:*** They express higher-level competencies enabled by the CR through the combined realization of decisional, functional, and informational aspects. Rather than referring to individual components or services, capabilities designate what the CR is able to support (e.g., attack simulation, defensive training, or incident response evaluation).

Note that, from a modeling perspective, the ontology is formulated as a domain-specific artifact, i.e., focusing on the CR domain rather than on task-specific or application-specific processes. In this context, it emphasizes: (i) clear conceptual boundaries between different CR aspect categories, (ii) semantic relationships between them, and (iii) minimal coupling to implementation technologies. Note that the above-discussed design choices are deliberate, as they facilitate the subsequent instantiation of the model in OWL using Protégé, where classes, object properties, and constraints can be directly derived from the conceptual entities introduced in the modeling phase. Consequently, each of these entities, later defined in section 4.2, is formulated with a one-to-one or one-to-few mapping to ontology primitives (e.g. classes, object properties, and constraints), ensuring coherence between the conceptual model introduced in this section and its formal implementation in the following section 5. Summarizing, the model establishes the basis for machine-supported reasoning and explainable design and evaluation processes; aptitudes that are otherwise difficult to achieve using purely textual, descriptive, or diagrammatic representations.

4.2. Model Design

This subsection outlines the CR ontology model in terms of (i) the class hierarchy, (ii) object and datatype properties, and (iii) axioms that capture core constraints and enable reasoning. The model is defined to align with the classification of Figure 1 and the modeling scope as defined in section 4.1, i.e., it represents how the vertical classification of infrastructure-decisional and scope-decisional aspects, as well as functional and informational aspects relates to the classification's horizontal – broad aspects of the CR, core infrastructure, RLMS, and user-oriented – layers. Figure 2 provides a high-level, consolidated visual representation of the proposed CR ontology. The upper part of the figure presents the high-level conceptual structure, where the `CyberRange` entity acts as the central aggregation point for decisional, functional, and informational aspects, as well as for the capabilities supported.

The direction of the arrows is intentional, aiming to convey the role of each category. Specifically, decisional aspects are modeled as pointing towards the `CyberRange`, reflecting that they represent external design-time choices and constraints that shape and define the CR before its realization. In contrast, functional and informational aspects are modeled as emanating from the `CyberRange`, as they describe operational behaviors and data artifacts that are instantiated and produced by the CR at engineering and runtime phases. Another interpretation for the arrow direction is that decisional aspects represent upstream constraints in the ontology, semantically governing the instantiation of functional and informational aspects and, by extension, the capabilities supported by the CR.

 doi:10.20944/preprints202602.2048.v1

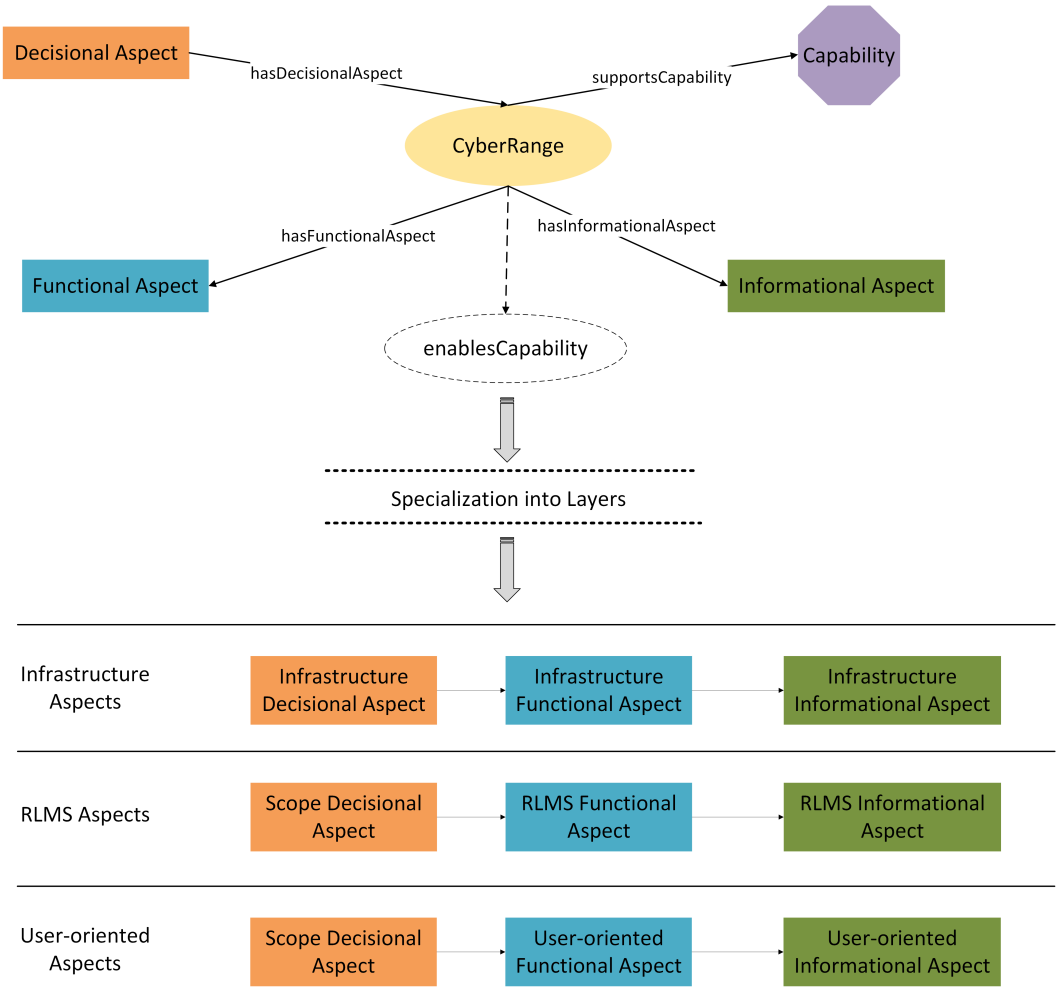


Figure 2. A bird's eye view of the proposed CR ontology.

The lower part of the figure depicts how these aspects are systematically specialized across the three horizontal layers of a CR ecosystem, namely the infrastructure, RLMS, and user-oriented layers; the fourth horizontal class of figure 1, i.e., the broad aspects of the CR, is not depicted as a separate specialization. In the ontology, these aspects are already captured within the decisional aspects and directly associated with the `CyberRange` entity. Introducing an additional specialization would therefore duplicate decisional constructs without adding semantic differentiation, while disrupting the orthogonality between horizontal layers and vertical aspect types. This layered refinement illustrates the alignment between the ontology and the quad-level CR classification, showing how each aspect type is consistently instantiated at different architectural levels. Together, the two parts of the figure bridge high-level conceptual modeling and layered architectural realization, clarifying how design-time intent propagates to operational functionality, information production, and ultimately to supported CR capabilities.

4.2.1. Classes (Concepts)

At the core of the proposed ontology are the classes that represent the building blocks of a CR ecosystem. Each class corresponds to a fundamental entity that supports the design, implementation, or operation of a CR. The class hierarchy is designed to be both expressive and modular, enabling semantic clarity and reuse across diverse CR types. In this context, the ontology follows a core abstraction as follows: it introduces a root class, `CyberRange`, which acts as the container entity for each modeled CR instance. This class is semantically enriched by three major abstract categories that capture the essential dimensions of CR design, as summarized below.

 doi:10.20944/preprints202602.2048.v1

- `DecisionalAspect`: Represents high-level architectural and strategic decisions made during CR conception, such as technology selection or CR focus.
- `FunctionalAspect`: Encapsulates runtime behaviors, services, and technical modules that operationalize the CR's objectives (e.g., monitoring, automation).
- `InformationalAspect`: Covers the data-related artifacts involved in the CR, including configurations, telemetry, logs, or learning content.

To ensure fidelity with the layered structure of CR ecosystems, the model further divides both `FunctionalAspect` and `InformationalAspect` into three layer-specific subclasses, matching the infrastructure, RLMS, and user-oriented groups of aspects, as depicted in figure 2. This layered modeling enables precise annotation of which part of the CR stack is responsible for a given function or produces a particular type of data.

- `InfrastructureFunctionalAspect`, `RLMSFunctionalAspect`, `UserOrientedFunctionalAspect`
- `InfrastructureInformationalAspect`, `RLMSInformationalAspect`, `UserOrientedInformationalAspect`

In parallel, `DecisionalAspect` is divided into two subclasses that reflect the two primary domains of CR planning. These distinctions ensure that each design-time decision is associated with the appropriate architectural or instructional concern.

- `InfrastructureDecisionalAspect`: Includes choices like virtualization technology, infrastructure type, or delivery model.
- `ScopeDecisionalAspect`: Covers decisions regarding pedagogical strategies, user types, curricula, and evaluation methods.

For each sub-dimension of the decisional aspects (e.g., governance model, learning method), the ontology defines controlled (dedicated) classes that act as vocabularies. For instance, the class `Governance` represents possible governance structures (e.g., autonomous, federated), while `Learning Method` captures different instructional paradigms (e.g., self-paced, instructor-led). These controlled vocabularies will later be populated with OWL individuals in the validation phase, detailed in section 6.

The ontology also introduces the class `Capability`, which is used to describe what the CR is ultimately able to support, including attack simulation, incident response evaluation, or Capture-the-Flag (CtF) exercises. Unlike lower-level aspects, `Capability` entities emerge from the composition of decisional, functional, and informational elements. As such, they play a central role in linking design intent to platform resources and serve as a useful anchors for reasoning and comparison across CRs. Finally, to enable concrete mapping of functions and services to technical modules or actors, the ontology includes two grounding classes. These grounding entities are intended to bridge the abstract model with concrete platform instantiations and usage scenarios.

- `Component`: A generic abstraction for tangible software or infrastructure elements, further specialized as `InfrastructureComponent`, `RLMSComponent`, and `UserOrientedComponent`.
- `UserEntity`: Represents human actors or groups participating in the CR, with common roles such as `Trainee`, `Instructor`, and `Team`.

4.2.2. Properties (Relationships)

So far, classes define the conceptual entities of the ontology. Consequently, this subsection introduces the properties that delineate how they are semantically connected. In the proposed CR ontology model, properties establish structural dependencies between classes, capture descriptive annotations, and enable logical reasoning over CR configurations. Two main categories of properties are introduced: object and datatype properties. All object and datatype properties are defined with explicit domain and range constraints in the OWL implementation, later detailed in section 5, ensuring semantic consistency and enabling automated validation in Protégé. Collectively, they transform

 doi:10.20944/preprints202602.2048.v1

the ontology from a static classification scheme into a structured, machine-interpretable model that supports reasoning over CR design decisions, operational behavior, and capability realization.

Object Properties. Object properties describe relationships between instances of ontology classes. Namely, they formulate the semantic backbone of the model, allowing a `CyberRange` to be represented as a structured and interrelated ecosystem rather than as an isolated conceptual entity.

At the highest level, aggregation properties link a `CyberRange` to its defining elements. The properties `hasDecisionalAspect`, `hasFunctionalAspect`, and `hasInformationalAspect` associate a CR instance with the respective aspects that characterize its design and operation, as illustrated in Figure 2. Similarly, `supportsCapability` connects a `CyberRange` to the `Capability` concepts it ultimately enables. For reasoning completeness, inverse properties (e.g., `isDecisionalAspectOf`, `isFunctionalAspectOf`, `isInformationalAspectOf`) are defined to facilitate bidirectional navigation and querying in Protégé, later used in the ontology implementation phase in section 5. Beyond aggregation, the ontology introduces dependency properties that reflect the internal logic of CR design. The properties `constrainsFunction`, `constrainsInformation`, and `constrainsUsers` model how decisional aspects shape the functional and informational and users' realization and participation in the CR, respectively. For example, a specific delivery model may restrict the orchestration mechanisms that can be deployed. At the same time, a particular learning method may determine which informational artifacts must be generated for evaluation purposes.

To ensure that controlled vocabulary classes are semantically integrated into the ontology, specialization properties are introduced to connect decisional aspects with their corresponding sub-dimensions. For infrastructure-decisional aspects, properties such as `hasGovernance`, `hasDeliveryModel`, `hasInfrastructureType`, `hasVirtualizationTechnology`, `hasDomain`, and `hasScope` link the abstract decisional entity to the appropriate controlled vocabulary classes. Similarly, scope-decisional aspects are connected through properties such as `hasLearningMethod`, `hasEvaluationPractice`, `hasCurriculaType`, `hasPurpose`, `targetsDidactics`, `definesUserType`, and `definesUserTeam`. These relations ensure that decisional knowledge is not merely taxonomical but explicitly instantiated and queryable within the ontology. Runtime behavior and data flow are modeled through the properties `producesInformation` and `consumesInformation`, which define the interaction between functional and informational aspects. These relationships capture how specific functionalities generate logs, telemetry, learning content, or evaluation results, and how they consume configuration artifacts or user inputs. Inverse counterparts (e.g., `isProducedBy`) are defined to enhance traceability and consistency in reasoning.

Capability realization is expressed through the property `enablesCapability`, linking functional or informational aspects to the `Capability` concepts they contribute to. Identically to aggregation properties, its inverse (e.g., `isSupportedByAspect`) strengthens interpretability during validation and reasoning. When combined with aggregation properties, this structure enables inference patterns whereby a `CyberRange` can be automatically classified as supporting a specific capability based on its declared aspects. Finally, grounding relations such as `hasComponent` and `hasUserEntity` connect abstract aspects to implementation modules and human actors. These properties link the conceptual model with tangible system elements and user roles, enhancing explainability and applicability. They allow functional aspects defined at a conceptual level (e.g., monitoring or orchestration) to be explicitly associated with the components that realize them within a specific CR deployment. Similarly, linking user-oriented aspects to user entities (e.g., trainees, instructors, or teams) enables the ontology to capture the structure of human participation within the CR. Altogether, this grounding ensures that the ontology remains not only theoretically coherent but also directly mappable to real CR implementations and operational configurations.

Datatype Properties. In contrast to object properties, datatype properties connect ontology instances to literal values (e.g., strings, identifiers, or numeric attributes). They are primarily used for descriptive annotations rather than structural reasoning. Core metadata properties such as `hasName` and `hasDescription` provide human-readable labels and explanations for ontology entities. Versioning

 doi:10.20944/preprints202602.2048.v1

and traceability are supported through properties such as `hasVersion` and `hasID`. Additional optional attributes, including `hasPriority`, `hasLevel`, `hasTimestamp`, or `hasDuration`, may be used to annotate complexity, criticality, or temporal characteristics when required by specific use cases.

4.2.3. Axioms (Constraints)

While classes define the structural vocabulary of the ontology and properties define how entities are connected, axioms provide the logical rules that govern their behavior. In the proposed CR ontology, axioms ensure semantic consistency, enforce modeling constraints, and enable automated reasoning within OWL-compliant tools such as Protégé. These constraints formalize the intended interpretation of the model and prevent ambiguous or inconsistent CR descriptions. Regarding disjointness axioms, to preserve conceptual clarity and avoid unintended overlaps between modeling dimensions, we define several disjointness constraints. The three primary aspect categories, `DecisionalAspect`, `FunctionalAspect`, and `InformationalAspect`, are declared pairwise disjoint. This enforces the vertical separation of concerns described in Section 3.1, ensuring that a given element cannot simultaneously represent a design decision, a runtime behavior, and a data artifact.

Similarly, layer-specific subclasses within each aspect pillar are declared mutually disjoint (e.g., `InfrastructureFunctionalAspect`, `RLMSFunctionalAspect`, and `UserOrientedFunctionalAspect`). This guarantees that each aspect instance belongs to exactly one architectural layer, maintaining alignment with the quad-level CR classification of figure 1. Additionally, the class `Capability` is defined as disjoint from all aspect and component classes, reinforcing its role as an emergent abstraction rather than a structural or operational element. For completeness and to avoid unintended classification overlap, the classes `Component` and `UserEntity` are also declared disjoint, ensuring a clear separation between technical implementation modules and human actors within the CR ecosystem.

Concerning domain and range restrictions, all object properties are defined with explicit domain and range constraints to ensure semantic correctness. For example, `hasDecisionalAspect` is restricted to link a `CyberRange` to instances of `DecisionalAspect`, while `producesInformation` connects instances of `FunctionalAspect` to `InformationalAspect`. Likewise, specialization properties such as `hasGovernance` or `hasLearningMethod` are constrained to associate decisional aspects with their corresponding controlled vocabulary classes. These domain and range specifications serve two purposes. First, they guide consistent ontology instantiation. Second, they allow reasoning engines to infer implicit class memberships. For instance, if an individual is linked via `producesInformation`, it can be inferred to belong to the class `FunctionalAspect`.

With respect to cardinality constraints, minimum cardinalities are introduced to ensure that a CR description is structurally complete. Each `CyberRange` must be associated with at least one decisional, one functional, and one informational aspect, reflecting the modeling assumption that a valid CR instance must incorporate design intent, operational behavior, and data manifestation. Similarly, each `Capability` must be enabled by at least one functional or informational aspect, formalizing the notion that capabilities cannot exist independently of the mechanisms that realize them. Where appropriate, maximum cardinality or functional property constraints are additionally applied. In particular, infrastructure decisional specialization properties are modeled as functional when only a single value is logically admissible per CR instance (e.g., `hasDeliveryModel`, `hasVirtualizationTechnology`). Other properties remain non-functional to preserve modeling flexibility in cases where multiple values may be valid. It should be noted that these cardinality constraints ensure minimum structural completeness while remaining compatible with OWL's semantics, meaning that the absence of explicitly asserted relationships does not imply their non-existence.

To enhance navigability and reasoning expressiveness, several object properties are defined with explicit inverse counterparts. For example, `hasDecisionalAspect` is paired with `isDecisionalAspectOf`, and `producesInformation` with `consumesInformation`. These inverse relations enable bidirectional querying and strengthen traceability within CR descriptions. Another central reasoning mechanism of the ontology refers to the inference of supported capabilities from declared aspects. For example, the following is formalized through a property-chain axiom 1.

 doi:10.20944/preprints202602.2048.v1

$$\texttt{hasFunctionalAspect} \circ \texttt{enablesCapability} \Rightarrow \texttt{supportsCapability} \quad (1)$$

This rule states that if a `CyberRange` has a functional aspect that enables a specific capability, then the CR is inferred to support that capability. Similar reasoning patterns can be applied to informational aspects. Through such axioms, the ontology supports automated capability classification, comparability across CR instances, and explainable validation of CR configurations. Together, these axioms transform the ontology from a descriptive schema into a logically grounded knowledge model. They ensure that CR descriptions remain consistent, complete, and semantically interpretable, while enabling automated reasoning over design decisions, operational characteristics, and emergent capabilities.

4.2.4. Summary of Ontology Elements

Table 2 recapitulates the structural elements of the proposed CR ontology, grouping them into classes, object and datatype properties, and axioms that define its semantic backbone. The class hierarchy reflects the quad-level CR classification, distinguishing decisional, functional, and informational aspects, while preserving their specialization across infrastructure, RLMS, and user-oriented layers.

Table 2. Summary of Core Ontology Elements for the Cyber Range Ontology Model

Element Type	Name	Description
Classes (Concepts)		
Class	CyberRange	Root entity representing a complete CR instance
Class	DecisionalAspect	Abstract class for design-time decisions
Class	FunctionalAspect	Abstract class for runtime behaviors and services
Class	InformationalAspect	Abstract class for data artifacts and information objects
Class	Capability	Emergent competencies enabled by the CR
Class	Component	Abstract technical module implementing functional aspects
Class	UserEntity	Human actors participating in CR activities
Subclass	InfrastructureDecisionalAspect	Infrastructure-related design decisions
Subclass	ScopeDecisionalAspect	Pedagogical and audience-oriented decisions
Subclass	InfrastructureFunctionalAspect	Infrastructure-layer functionalities
Subclass	RLMSFunctionalAspect	RLMS-layer functionalities
Subclass	UserOrientedFunctionalAspect	User-facing services and mechanisms
Subclass	InfrastructureInformationalAspect	Infrastructure-generated data artifacts
Subclass	RLMSInformationalAspect	RLMS-related informational artifacts
Subclass	UserOrientedInformationalAspect	User-facing or user-generated artifacts
Subclass	InfrastructureComponent, RLMSComponent, UserOrientedComponent	Layer-specific component implementations
Controlled Classes	Governance, DeliveryModel, InfrastructureType, VirtualizationTechnology, Domain, Scope	Infrastructure decisional vocabularies
Controlled Classes	LearningMethod, EvaluationPractice, CurriculumType, Purpose, Didactics, UserType, UserTeam	Scope decisional vocabularies
Object Properties		
Aggregation	hasDecisionalAspect / isDecisionalAspectOf	Links CR to decisional aspects (inverse defined)
Aggregation	hasFunctionalAspect / isFunctionalAspectOf	Links CR to functional aspects (inverse defined)
Aggregation	hasInformationalAspect / isInformationalAspectOf	Links CR to informational aspects (inverse defined)
Aggregation	supportsCapability	Declares capabilities supported by a CR
Dependency	constrainsFunction	DecisionalAspect → FunctionalAspect
Dependency	constrainsInformation	DecisionalAspect → InformationalAspect
Dependency	constrainsUsers	DecisionalAspect → UserEntity
Runtime Flow	producesInformation / consumesInformation	FunctionalAspect → InformationalAspect (inverse defined)
Capability Realization	enablesCapability / isSupportedByAspect	Functional/InformationalAspect → Capability (inverse defined)
Grounding	hasComponent	Links FunctionalAspect to implementing Component
Grounding	hasUserEntity	Associates aspects or CR with UserEntity
Specialization	hasGovernance, hasDeliveryModel, hasInfrastructureType, hasVirtualizationTechnology, hasDomain, hasScope	Infrastructure decisional specialization
Specialization	hasLearningMethod, hasEvaluationPractice, hasCurriculaType, hasPurpose, targetsLearningObjective, targetsDidactics, definesUserType, definesUserTeam	Scope decisional specialization
Datatype Properties		
Metadata	hasName, hasDescription	Human-readable annotations
Traceability	hasID, hasVersion	Identification and version control
Annotation	hasPriority, hasLevel	Complexity or criticality attributes
Optional	hasTimestamp, hasDuration	Temporal annotations when required
Axioms and Constraints		
Disjointness	Aspect Classes Disjoint	$DecisionalAspect \perp FunctionalAspect \perp InformationalAspect$
Disjointness	Layer Subclasses Disjoint	Infrastructure ⊥ RLMS ⊥ UserOriented subclasses
Disjointness	Capability Disjointness	$Capability \perp$ Aspect ⊥ Component; Component ⊥ UserEntity
Domain/Range	Property Restrictions	Explicit domain and range constraints for all object properties
Cardinality	Structural Completeness	Each CR has ≥ 1 decisional, functional, informational aspect
Cardinality	Capability Support	Each Capability enabled by ≥ 1 aspect
Inverse Properties	Bidirectional Reasoning	$producesInformation \equiv consumesInformation^{-1}$
Property Chain	Capability Inference	$hasFunctionalAspect \circ enablesCapability \Rightarrow supportsCapability$

A deliberate modeling choice concerns the treatment of leaf elements of the classification of figure 1. Decisional leaves (e.g., `Governance`, `Purpose`, `Didactics`) are represented as controlled vocabulary

 doi:10.20944/preprints202602.2048.v1

classes, since they correspond to categorical design-time selections that are relatively stable, comparable across CR implementations, and suitable for structured reasoning. In contrast, functional and informational leaves (e.g., orchestration modules, scenario development mechanisms, interfaces) are instantiated as individuals under the respective aspect subclasses (e.g., `InfrastructureFunctionalAspect`, `RLMSFunctionalAspect`, `UserOrientedFunctionalAspect`, and their informational counterparts) or, where structurally appropriate, modeled as extensible subclasses within these layer-specific categories. This design preserves alignment with the baseline classification while maintaining flexibility, allowing diverse CR implementations to introduce platform-specific modules or data artifacts without modifying the ontology schema. The object properties listed in the table capture aggregation, dependency, specialization, runtime data flow, capability realization, and grounding relations, enabling traceability from design-time decisions to operational functionality and emergent capabilities. Finally, the axioms enforce disjointness, domain and range constraints, cardinality conditions, inverse relations, and capability inference patterns, ensuring logical consistency and supporting automated reasoning within Protégé.

In addition to the structural overview provided in Table 2, Figure 3 offers a consolidated visual representation of the ontology's core elements and their interrelations through the corresponding object properties. While Table 2 enumerates the classes, properties, and axioms in a structured textual form, the figure illustrates how these elements are instantiated within the modeling framework introduced earlier through subsections 4.1 to 4.2 and recapitulated in table 2. Essentially, Figure 3 complements the tabular summary by visually illustrating how the listed classes and object properties are organized and connected within the ontology. In other words, it depicts the `CyberRange` entity at the center and its explicit links to decisional, functional, and informational aspects, their layer-specific specializations, and the associated capabilities, rather than presenting the elements in isolation, as summarized in the table.

5. Implementation

The CR ontology introduced in section 4 was implemented in OWL 2 DL using Protégé. The conceptual model was translated into formal OWL classes, object properties, datatype properties, and logical axioms. The ontology was serialized in RDF/XML format to ensure interoperability with standard Semantic Web tools and reasoning engines. All primary classes of table 2 were declared as OWL classes. Layer-specific specializations were implemented using `rdfs:subClassOf` relationships, preserving the orthogonal vertical–horizontal structuring of the model. Controlled vocabulary elements corresponding to decisional leaves (e.g., DeliveryModel, LearningMethod, Governance) were modeled as dedicated subclasses under their respective decisional branches. Grounding classes for technical artifacts and human actors were implemented through the use of Component and UserEntity specializations. Figure 4(**a**) presents the inferred class hierarchy as visualized in Protégé. The figure confirms the structuring of vertical abstraction pillars and their horizontal refinements across infrastructure, RLMS, and user-oriented layers.

Object properties were defined with explicit domain and range constraints to enforce semantic correctness. Aggregation, dependency, runtime flow, capability realization, and specialization properties were implemented according to the design summarized in Table 2. Figure 4(**b**) illustrates the object property hierarchy in Protégé, showing the complete set of relationships supporting CR aggregation and capability inference. Logical constraints were encoded using OWL disjointness axioms, qualified minimum cardinality restrictions for structural completeness of `CyberRange` instances, and functional object properties for infrastructure decisional specialization. The ontology was validated using the HermiT 1.4.3.456 reasoner in Protégé, confirming logical consistency and correct enforcement of modeling constraints, as shown in 5. At this point, it is important to notice that this reasoning capability is one of the principal advantages of the proposed ontology. Namely, unlike traditional architectural representations, which, as the reader may recall from section 1, remain descriptive and require manual interpretation, the OWL 2 DL formalization enables automated logical verification and inference over

 doi:10.20944/preprints202602.2048.v1

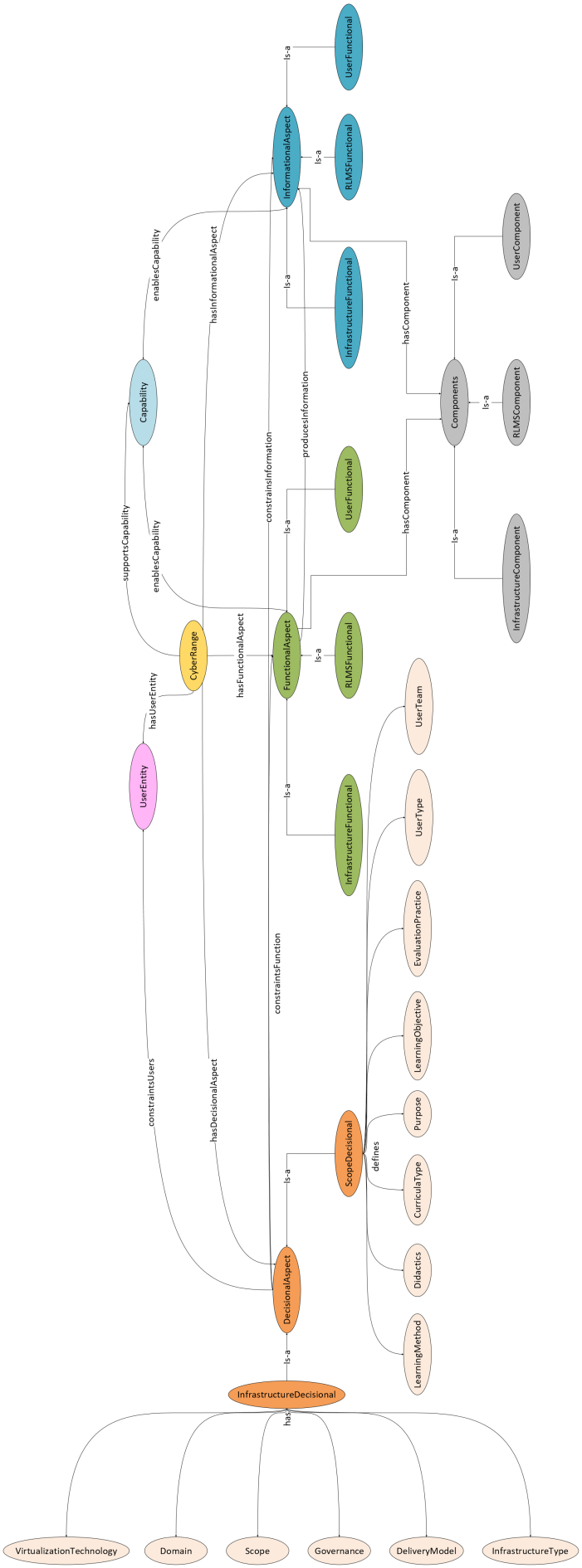
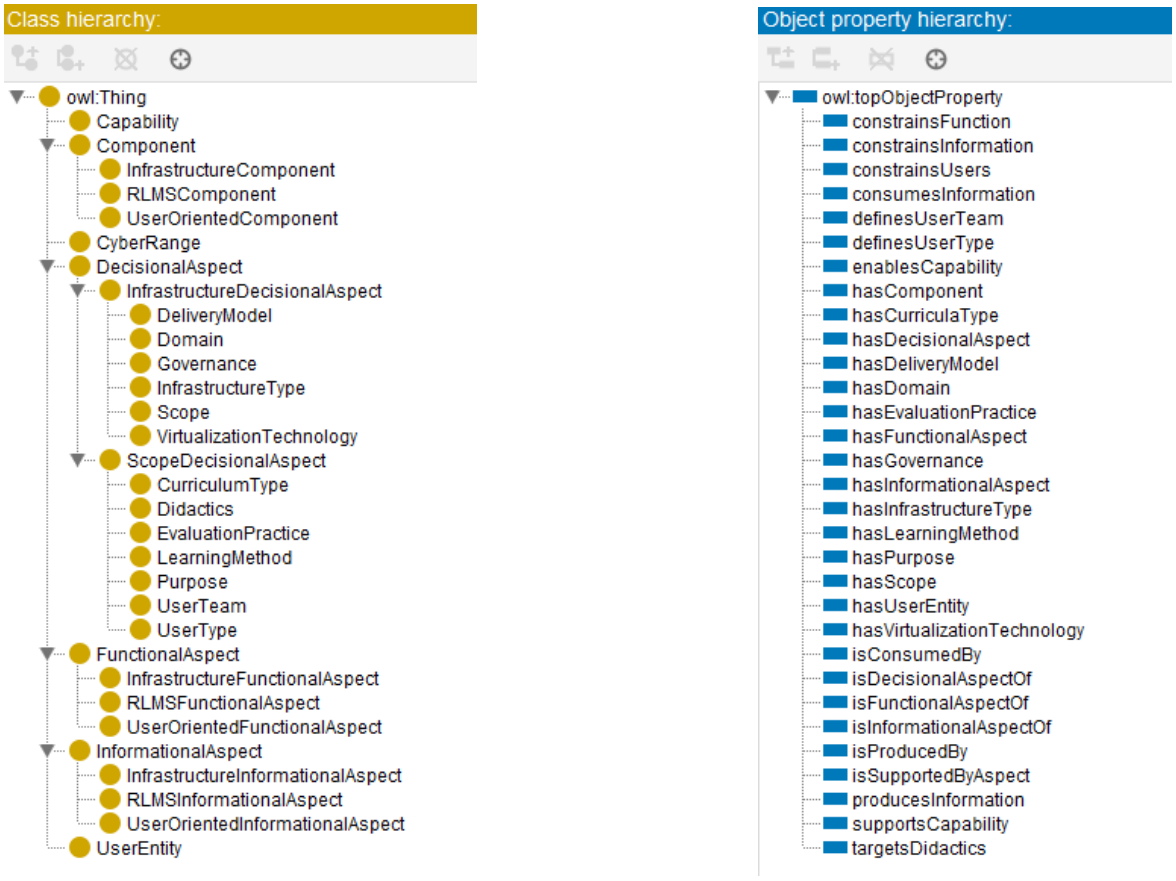


Figure 3. A consolidated overview of the proposed CR ontology.

 doi:10.20944/preprints202602.2048.v1



(**a**) Inferred class hierarchy. (**b**) Object property hierarchy.

Figure 4. Implementation of the CR ontology in Protégé.

CR configurations, beyond just structured querying (e.g., SPARQL queries). Specifically, reasoning provides the following capabilities.

- ***Consistency checking:*** The reasoner automatically detects violations of disjointness axioms, domain and range constraints, and cardinality restrictions.
- ***Structural completeness:*** The reasoner verifies that each `CyberRange` instance satisfies minimum cardinality requirements, ensuring the presence of decisional, functional, and informational aspects as mandated by the ontology model introduced in section 4.
- ***Implicit classification:*** The reasoner infers class memberships based on asserted object property relations and their declared domain and range constraints.
- ***Capability inference:*** The reasoner applies property-chain axiom 1 to propagate enabled capabilities from functional aspects to the corresponding `CyberRange` instance.
- ***Constraint enforcement:*** The reasoner ensures that decisional aspects semantically constrain admissible functional and informational realizations, maintaining coherence between design-time decisions and runtime behavior.

These reasoning mechanisms elevate the ontology from a structured vocabulary to a logically grounded validation and inference framework capable of supporting explainable CR configuration analysis. Note that, at this stage, reasoning operates exclusively at the schema level (i.e., over classes and axioms), since no CR instances have yet been introduced. Consequently, the reasoner performs ontology classification and constraint verification but does not produce instance-level inferences. More expressive reasoning outcomes, such as automatic capability inference for specific CR configurations, require the introduction of individuals and are demonstrated in the validation phase, presented later in Section 6.

 doi:10.20944/preprints202602.2048.v1

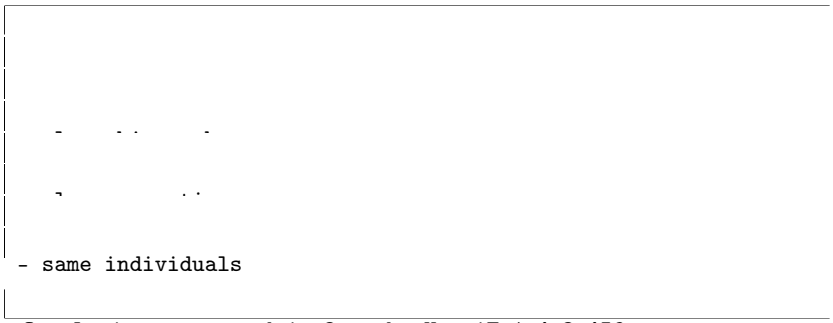


Figure 5. Reasoner execution log in Protégé using HermiT 1.4.3.456.

From a technical standpoint, the ontology conforms to the OWL 2 DL, ensuring decidability and compatibility with standard DL reasoners. The use of qualified cardinality restrictions, inverse object properties, functional properties, and property-chain axioms remains within the expressive bounds of SROIQ(D) [30], allowing complete reasoning under HermiT without compromising computational tractability. The ontology was exported in RDF/XML syntax using the OWL Application Programming Interface (API) integrated in Protégé, enabling straightforward reuse, extension, and integration with external Semantic Web frameworks. No custom rules, Semantic Web Rule Language (SWRL) expressions, or non-standard extensions were introduced, ensuring portability and compliance with standards.

6. Validation

To validate the expressive adequacy and reasoning capabilities of the proposed CR ontology, a CR platform, namely KYPO [17], was instantiated as an ontology individual. The objective of this validation phase is threefold: (i) to demonstrate structural completeness of the model at the instance level, (ii) to verify automated reasoning over capability realization, and (iii) to evaluate the ontology using representative competency questions expressed in the SPARQL protocol.

As presented in [17], KYPO is an open-source cyber range platform designed for hands-on cybersecurity training and cyber defense exercises. Architecturally, it relies on virtualized infrastructure resources for sandbox provisioning and isolation, enabling the deployment of reproducible training environments. The platform incorporates mechanisms for scenario definition and lifecycle management, automated orchestration of training instances, remote user access through a web-based interface, and collection of telemetry and evaluation data. Its design supports multi-user and multi-team execution modes, facilitating both individual and collaborative exercises. Note that KYPO was selected for validation purposes due to its high citation impact in the CR literature, comprehensive architectural documentation [17,31–35], and well-defined modular design, which collectively provide sufficient publicly available technical detail to support ontology instantiation.

Figure 6 illustrates the object property assertions of the instantiated individual `KYPO_CyberRangePlatform` as visualized in Protégé after reasoning. The upper portion of the panel displays the explicitly asserted relations linking the CR instance to its decisional, functional, and informational aspects. In contrast, the `supportsCapability` relations highlighted in yellow correspond to inferred object property assertions generated by the reasoner. These inferred relations are not directly asserted in the ontology but are derived from the composition of `hasFunctionalAspect` and `enablesCapability`, as formalized by the property-chain axiom 1. The figure therefore provides direct evidence of successful ABox-level [36] reasoning and confirms that capability realization is correctly propagated to the CR instance level. To further assess the ontology's queryability and semantic coherence, three representative Competency Questions (CQ) were formulated and evaluated using SPARQL queries executed within Protégé, as shown in figures 7 to 9.

 doi:10.20944/preprints202602.2048.v1

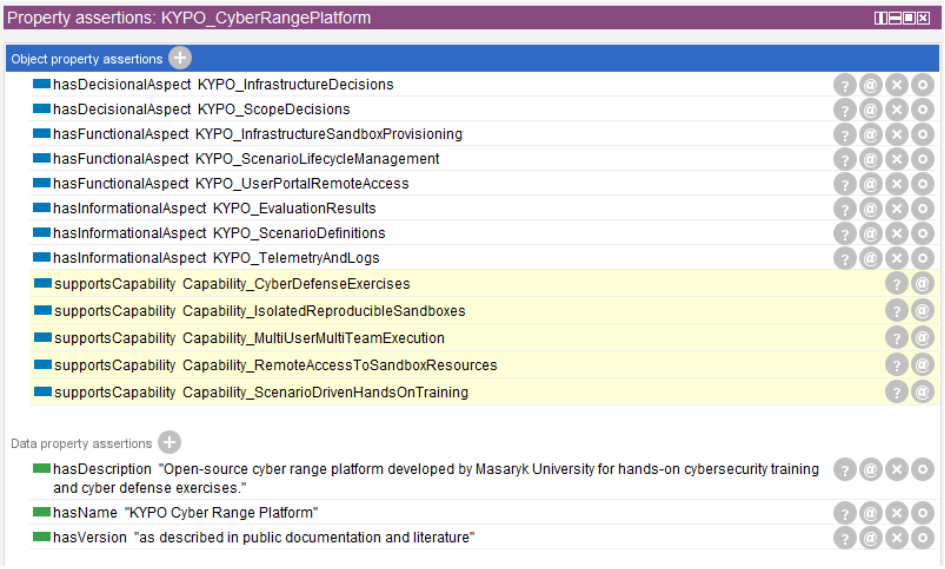


Figure 6. Object property assertions of `KYPO_CyberRangePlatform` in Protégé. Inferred `supportsCapability` relations are highlighted in yellow.

CQ1. Which functional aspects are associated with the instantiated CR platform? The results shown in Figure 7 confirm that the instantiated CR is associated with three functional aspects, corresponding to infrastructure-level sandbox provisioning, scenario lifecycle management, and user-oriented remote access mechanisms. The query outcome verifies the structural completeness of the instantiation with respect to the functional dimension of the ontology, demonstrating that functional aspects are explicitly retrievable through object property assertions. This confirms that the aggregation relation `hasFunctionalAspect` is correctly instantiated and queryable, enabling transparent inspection of the operational capabilities embedded within the CR configuration.

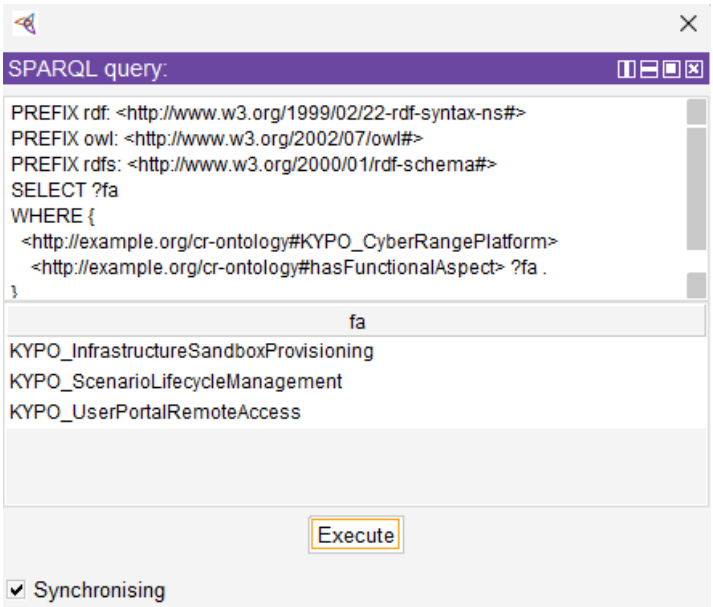


Figure 7. SPARQL result for CQ1: functional aspects of KYPO.

CQ2. Which capabilities are enabled by each functional aspect? The results depicted in Figure 8 reveal the mapping between functional aspects and the capabilities they enable. The query confirms that each functional aspect contributes to one or more higher-level competencies, thereby enacting the linkage between runtime behavior and emergent CR capabilities. This mapping illustrates the internal traceability supported by the ontology, demonstrating how operational mechanisms realize distinct training and execution capabilities. The outcome validates the correct instantiation of the

 doi:10.20944/preprints202602.2048.v1

`enablesCapability` relation and provides the logical foundation for the subsequent inference of CR-level supported capabilities.

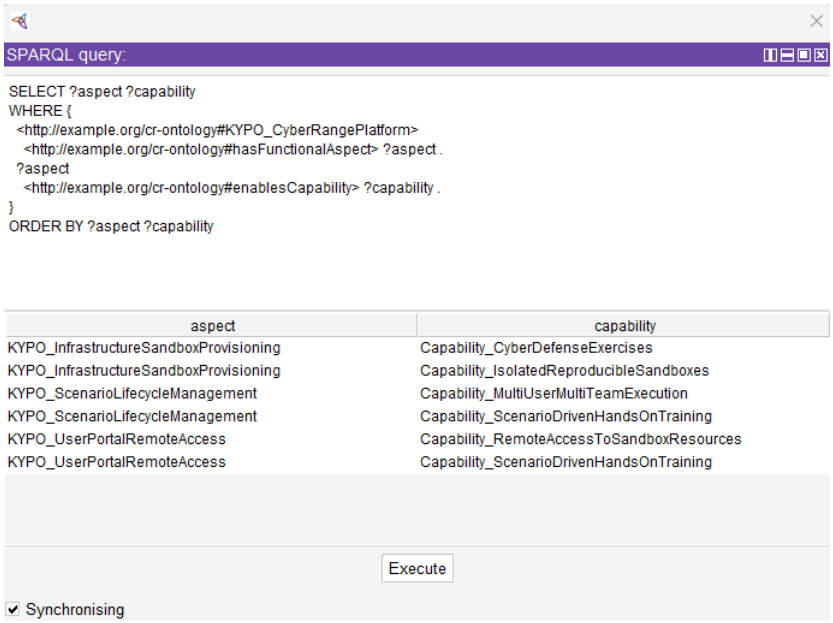


Figure 8. SPARQL result for CQ2: mapping between functional aspects and enabled capabilities.

CQ3. Which capabilities are supported by KYPO? The results shown in Figure 9 present the reconstructed set of capabilities supported by the instantiated KYPO platform. Although the `supportsCapability` relation is not explicitly asserted in the RDF graph, the query faithfully reproduces the entailment defined by the property-chain axiom 1 through the composition of `hasFunctionalAspect` and `enablesCapability`. The returned capabilities therefore correspond to those inferred by the reasoner at the ABox level, confirming semantic consistency between logical inference and query-based retrieval. This outcome demonstrates that the ontology enables capability-level assessment of a CR configuration and supports transparent traceability from operational mechanisms to emergent competencies. A direct SPARQL query over `supportsCapability` returned no results, as this relation is inferred by the reasoner and not materialized in the asserted RDF graph. Therefore, the competency question was reformulated according to the entailment pattern of the property-chain axiom 1:

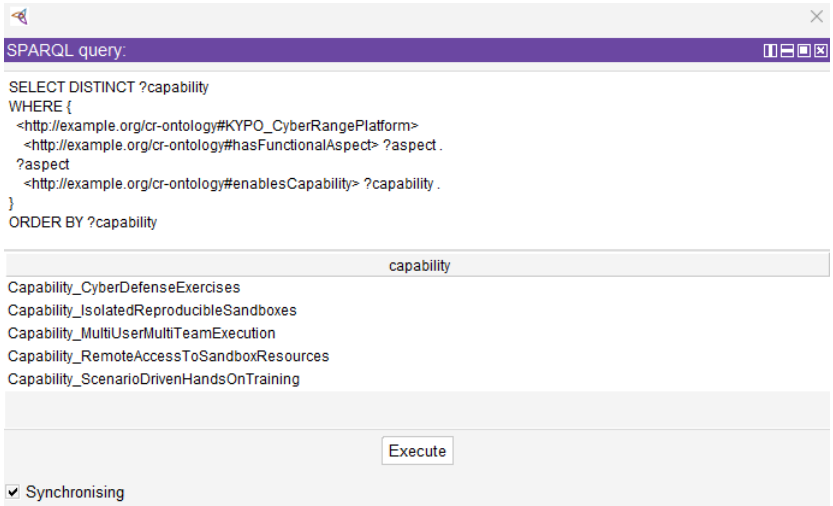


Figure 9. SPARQL result for CQ3: reconstructed supported capabilities of KYPO.

 doi:10.20944/preprints202602.2048.v1

Overall, the validation demonstrates that the proposed ontology supports the consistent instantiation of CR platforms while preserving logical soundness under OWL 2 DL semantics. The successful execution of the HermiT reasoner confirms that structural constraints and disjointness axioms are maintained at both schema and instance levels. Moreover, capability support is automatically inferred through the defined property-chain axioms, validating the intended reasoning mechanisms for propagating operational functionality to CR-level competencies. The formulation and execution of competency questions via SPARQL further illustrate that the ontology correctly enables the structured retrieval of design-time and runtime knowledge. In other words, these findings confirm that the proposed model operates not merely as a static classification scheme, but as a logically grounded, machine-interpretable knowledge framework capable of supporting CR configuration analysis, traceability across abstraction layers, and capability-based evaluation.

7. Limitations & Future Work

The proposed ontology provides a formally grounded and semantically coherent representation of CR ecosystems; however, several limitations must be acknowledged, and corresponding directions for future research are outlined to guide its further extension and refinement.

- ***Abstraction Level.*** The ontology is conceptualized and instantiated at a domain abstraction level and does not capture low-level implementation details of specific CR platforms. Namely, it models architectural, functional, informational, and decisional aspects in a technology-agnostic manner. Consequently, highly platform-specific configurations, deployment scripts, infrastructure-as-code artifacts, or detailed runtime orhestration logic are intentionally abstracted. This design choice enhances generalizability and cross-platform comparability, but it limits direct operational automation or platform-level configuration assembly without additional extensions. In this context, future research may focus on extending the ontology with optional implementation-level modules that align CR functional aspects with concrete deployment descriptors (e.g., container orchestration manifests, virtualization templates, or network configuration schemas). Such extensions could facilitate automation of CR deployment workflows while preserving the abstraction principles of the existing ontology.
- ***Expressiveness Boundaries.*** Although the ontology conforms to the OWL 2 DL, supporting decidable reasoning, its expressive power is bounded by DL semantics. That is, complex procedural constraints, dynamic workflows, temporal sequencing of CR activities, or conditional logic between events beyond the native representational scope of DL-based ontologies. Such behaviors would require rule-based extensions (e.g., SWRL) or integration with process-oriented formalisms. Future extensions may investigate the integration of rule-based reasoning mechanisms or alignment with process modeling frameworks (e.g., BPMN-derived ontologies or temporal logics) to represent dynamic CR workflows and event-driven behavior.
- ***Absence of Quantitative Evaluation Modeling.*** The ontology emphasizes structural and semantic relationships rather than quantitative evaluation metrics. Performance indicators, scalability measurements, resource utilization metrics, or detailed learning analytics are not formally encoded beyond optional datatype annotations. As a result, quantitative assessment of CR effectiveness remains outside the current modeling scope. Future research may incorporate dedicated evaluation and performance modeling modules, potentially through alignment with metric-oriented ontologies, learning analytics vocabularies, or evaluation formulas, such as the ones introduced in [14]. Such extensions would help the formal representation of CR attributes and support maturity assessment and benchmarking across heterogeneous platforms.
- ***Structural Capability Inference.*** Capability inference is currently based on structural composition through property-chain axioms, as highlighted in the property-chain axiom 1. This mechanism supports automated classification and traceability, however, it does not evaluate the qualitative adequacy, performance level, or maturity of the supported capability. Therefore, the ontology determines whether a capability is structurally supported, but not the degree to which it is effectively

 doi:10.20944/preprints202602.2048.v1

realized. Future work may explore the incorporation of graded or weighted capability models, integrating qualitative assessment criteria or maturity-level ontologies. Such enhancements could allow reasoning not only about capability presence but also about capability quality, effectiveness, or operational readiness.

- ***Scope of Empirical Validation.*** While current validation confirms logical consistency and structural integrity through an illustrative case-study instantiation, broader empirical validation across heterogeneous CR deployments would further strengthen the versatility of the proposed model. Although the selected CR platform provides a representative and well-documented example, future validation efforts may involve instantiating the ontology across multiple CR platforms with diverse architectural paradigms and conducting comparative capability analyses.

8. Conclusions

This paper introduced an ontology-based framework for the formal representation of CR ecosystems. To the best of our knowledge, this is the first attempt to formally model a CR ecosystem using a domain ontology that represents its structural, functional, informational, and decisional aspects in a unified semantic framework. In contrast to the current state of affairs, where architectural descriptions are mostly semi-formal, our approach offers a semantically rich and machine-processable representation that not only encodes the structural makeup but also the design intent of CR platforms. The ontology is structured along two orthogonal dimensions: a vertical abstraction distinguishing decisional, functional, and informational aspects and emergent capabilities, and a horizontal specialization that spans infrastructure, RLMS, and user-oriented layers. This two-fold organization ensures that the ontology is consistent with widely accepted CR categorizations and maintains a clear and modular conceptualization. The model is implemented in OWL 2 DL using Protégé, incorporating domain and range constraints, disjointness axioms, cardinality restrictions, inverse properties, and property-chain rules, enabling logical validation and automated reasoning over CR configurations. The validation process was performed by instantiating a representative CR platform, verifying structural completeness, consistency under reasoning, and automated capability derivation. Furthermore, competency questions expressed in SPARQL confirmed the ontology's suitability for structured querying and traceability analysis across design-time and runtime dimensions. Overall, the results indicate that the proposed ontology transcends static classification and functions as a logically grounded knowledge framework for CR modeling. The proposed framework lays the foundation for systematic comparison and capability-based evaluation of heterogeneous CR implementations.

Author Contributions: Conceptualization, V.K.; methodology, V.K.; software, V.K.; validation, V.K.; writing—original draft preparation, V.K.; writing—review and editing, V.K, M.T, V.G, S.K.; supervision, V.G, S.K.; project administration, V.G, S.K.; funding acquisition, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Research Council of Norway through the SFI Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS) project no. 310105 and by the European Union through the Horizon 2020 project PERSEUS (Grant No. 101034240).

Data Availability Statement: The OWL file corresponding to the ontology implementation in Protégé is publicly available in an open-source GitHub repository (https://github.com/byrkam/CR-ontology). The repository includes the serialized RDF/XML ontology file and supporting resources necessary to reproduce the validation experiments presented in this study.

Acknowledgments: The authors would like to acknowledge the use of Grammarly for language refinement and stylistic improvement of the manuscript. The tool was used exclusively for grammar and readability enhancement; all technical content, conceptual development, modeling decisions, and scientific conclusions remain solely the responsibility of the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

 doi:10.20944/preprints202602.2048.v1

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
CI	Critical Infrastructures
CPS	Cyber-Physical Systems
CR	Cyber Ranges
CQ	Competency Questions
CtF	Capture-the-Flag
CyRIS	Cyber Range Instantiation System
DL	Description Logic
DSL	Domain-Specific Language
HitL	Humans-in-the-Loop
IT	Information Technology
IaC	Infrastructure-as-Code
LLMs	Large Language Models
OWL	Web Ontology Language
RAG	Retrieval-Augmented Generation
RDF	Resource Description Framework
RLMS	Range Learning and Management System
SDL	Scenario Definition Language
SWRL	Semantic Web Rule Language
XML	Extensible Markup Language

References

1. Makrakis, G.M.; Kolias, C.; Kambourakis, G.; Rieger, C.; Benjamin, J. Industrial and Critical Infrastructure Security: Technical Analysis of Real-Life Security Incidents. *IEEE Access* **2021**, *9*, 165295–165325. https://doi.org/10.1109/ACCESS.2021.3133348.
2. Kampourakis, V.; Gkioulos, V.; Katsikas, S. A systematic literature review on wireless security testbeds in the cyber-physical realm. *Computers & Security* **2023**, *133*, 103383. https://doi.org/10.1016/j.cose.2023.103383.
3. Falliere, N.; Murchu, L.O.; Chien, E.; et al. W32. stuxnet dossier. *White paper, symantec corp., security response* **2011**, *5*, 29.
4. Kozak, P.; Klaban, I.; Šlajs, T. Industroyer cyber-attacks on Ukraine's critical infrastructure. In Proceedings of the 2023 International Conference on Military Technologies (ICMT), 2023, pp. 1–6. https://doi.org/10.1109/ICMT58149.2023.10171308.
5. Salazar, L.; Castro, S.R.; Lozano, J.; Koneru, K.; Zambon, E.; Huang, B.; Baldick, R.; Krotofil, M.; Rojas, A.; Cardenas, A.A. A Tale of Two Industroyers: It was the Season of Darkness. In Proceedings of the 2024 IEEE Symposium on Security and Privacy (SP), 2024, pp. 312–330. https://doi.org/10.1109/SP54263.2024.00162.
6. Datta, P. Hannibal at the gates: Cyberwarfare & the Solarwinds sunburst hack. *Journal of Information Technology Teaching Cases* **2022**, *12*, 115–120.
7. Kampourakis, V.; Kavallieratos, G.; Gkioulos, V.; Katsikas, S. Cracks in the chain: A technical analysis of real-life supply chain security incidents. *Computers & Security* **2025**, *159*, 104673. https://doi.org/10.1016/j.cose.2025.104673.
8. Beerman, J.; Berent, D.; Falter, Z.; Bhunia, S. A Review of Colonial Pipeline Ransomware Attack. In Proceedings of the 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW), 2023, pp. 8–15. https://doi.org/10.1109/CCGridW59191.2023.00017.
9. Kampourakis, V.; Gkioulos, V.; Katsikas, S. A step-by-step definition of a reference architecture for cyber ranges. *Journal of Information Security and Applications* **2025**, *88*, 103917. https://doi.org/10.1016/j.jisa.2024.103917.
10. Kampourakis, V. Secure Infrastructure for Cyber-Physical Ranges. In Proceedings of the Research Challenges in Information Science: Information Science and the Connected World, Cham, 2023; pp. 622–631. https://doi.org/10.1007/978-3-031-33080-3_45.
11. Katsantonis, M.N.; Manikas, A.; Mavridis, I.; Gritzalis, D. Cyber range design framework for cyber security education and training. *International Journal of Information Security* **2023**, *22*, 1005–1027. https://doi.org/10.1007/s10207-023-00680-4.

 doi:10.20944/preprints202602.2048.v1

12. Yamin, M.M.; Katt, B.; Gkioulos, V. Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. *Computers & Security* **2020**, *88*, 101636. https://doi.org/https://doi.org/10.1016/j.cose.2019.101636.
13. Ukwandu, E.; Farah, M.A.B.; Hindy, H.; Brosset, D.; Kavallieros, D.; Atkinson, R.; Tachtatzis, C.; Bures, M.; Andonovic, I.; Bellekens, X. A Review of Cyber-Ranges and Test-Beds: Current and Future Trends. *Sensors* **2020**, *20*. https://doi.org/10.3390/s20247148.
14. Kampourakis, V.; Kavallieratos, G.; Spathoulas, G.; Gkioulos, V.; Katsikas, S. LLM-Assisted AHP for Explainable Cyber Range Evaluation. *arXiv preprint arXiv:2512.10487* **2025**.
15. Wen, S.F.; Yamin, M.M.; Katt, B. Ontology-Based Scenario Modeling for Cyber Security Exercise. In Proceedings of the 2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), 2021, pp. 249–258. https://doi.org/10.1109/EuroSPW54576.2021.00032.
16. University, S. Protégé: A free, open-source ontology editor and framework for building intelligent systems. https://protege.stanford.edu/, 2016.
17. Vykopal, J.; Oslejsek, R.; Celeda, P.; Vizvary, M.; Tovarnak, D. KYPO Cyber Range: Design and Use Cases. In Proceedings of the Proceedings of the 12th International Conference on Software Technologies - ICSOFT. INSTICC, SciTePress, 2017, pp. 310–321. https://doi.org/10.5220/0006428203100321.
18. Bernardinetti, G.; Iafrate, S.; Bianchi, G. Nautilus: A Tool For Automated Deployment And Sharing Of Cyber Range Scenarios. In Proceedings of the Proceedings of the 16th International Conference on Availability, Reliability and Security, New York, NY, USA, 2021; ARES '21. https://doi.org/10.1145/3465481.3469182.
19. Pham, C.; Tang, D.; Chinen, K.i.; Beuran, R. CyRIS: a cyber range instantiation system for facilitating security training. In Proceedings of the Proceedings of the 7th Symposium on Information and Communication Technology, New York, NY, USA, 2016; SoICT '16, p. 251–258. https://doi.org/10.1145/3011077.3011087.
20. Gallus, P.; Klaban, I.; Františ, P.; Šlajs, T. Modular Cybersecurity Scenario Development and Its Integration into the Cyber Range Platform. In Proceedings of the 2025 Communication and Information Technologies (KIT), 2025, pp. 1–6. https://doi.org/10.1109/KIT67756.2025.11205464.
21. OWASP. OWASP Top Ten 2025. https://owasp.org/www-project-top-ten/, 2025. Accessed: 2025-11-24.
22. Russo, E.; Costa, G.; Armando, A. Building next generation Cyber Ranges with CRACK. *Computers & Security* **2020**, *95*, 101837. https://doi.org/10.1016/j.cose.2020.101837.
23. OASIS. OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) TC. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca, 2025. Accessed: 2025-11-24.
24. Gustafsson, T.; Almroth, J. Cyber Range Automation Overview with a Case Study of CRATE. In Proceedings of the Secure IT Systems; Asplund, M.; Nadjm-Tehrani, S., Eds., Cham, 2021; pp. 192–209. https://doi.org/10.1007/978-3-030-70852-8_12.
25. Yamin, M.M.; Katt, B. Modeling and executing cyber security exercise scenarios in cyber ranges. *Computers & Security* **2022**, *116*, 102635. https://doi.org/10.1016/j.cose.2022.102635.
26. Mudassar Yamin, M.; Hashmi, E.; Ullah, M.; Katt, B. Applications of LLMs for Generating Cyber Security Exercise Scenarios. *IEEE Access* **2024**, *12*, 143806–143822. https://doi.org/10.1109/ACCESS.2024.3468914.
27. Gruber, T.R. A translation approach to portable ontology specifications. *Knowledge Acquisition* **1993**, *5*, 199–220. https://doi.org/10.1006/knac.1993.1008.
28. Wand, Y.; Storey, V.C.; Weber, R. An ontological analysis of the relationship construct in conceptual modeling. *ACM Trans. Database Syst.* **1999**, *24*, 494–528. https://doi.org/10.1145/331983.331989.
29. Uschold, M.; Gruninger, M. Ontologies: principles, methods and applications. *The Knowledge Engineering Review* **1996**, *11*, 93–136. https://doi.org/10.1017/S0269888900007797.
30. Horrocks, I.; Kutz, O.; Sattler, U. The Even More Irresistible SROIQ. *Kr* **2006**, *6*, 57–67.
31. University, M. CyberRangeCZ Platform. https://docs.platform.cyberrange.cz/, 2020.
32. Sapák, T. kypo-crp-tf-deployment. https://gitlab.ics.muni.cz/muni-kypo-crp/devops/kypo-crp-tf-deployment, 2020.
33. Univeristy, M. KYPO CYBER RANGE PLATFORM. https://crp.kypo.muni.cz/, 2020.
34. Švábenský, V.; Vykopal, J.; Seda, P.; Čeleda, P. Dataset of shell commands used by participants of hands-on cybersecurity training. *Data in Brief* **2021**, *38*, 107398. https://doi.org/10.1016/j.dib.2021.107398.
35. Vykopal, J.; Vizvary, M.; Oslejsek, R.; Celeda, P.; Tovarnak, D. Lessons learned from complex hands-on defence exercises in a cyber range. In Proceedings of the 2017 IEEE Frontiers in Education Conference (FIE), 2017, pp. 1–8. https://doi.org/10.1109/FIE.2017.8190713.

 doi:10.20944/preprints202602.2048.v1

36. Makhlouf, A.; Percebois, C.; Tran, H.N. Two-Level Reasoning About Graph Transformation Programs. In Proceedings of the Graph Transformation; Guerra, E.; Orejas, F., Eds., Cham, 2019; pp. 111–127. https://doi.org/10.1007/978-3-030-23611-3_7.