Научная статья
УДК 004.056 + 005.21
https://doi.org/10.31854/1813-324X-2026-12-3-112-128
EDN:JIZXYX

Архитектурная модель ONYX для X-адаптивного управления информационной системой в условиях дестабилизирующих факторов произвольной природы

Виталий Владимирович Грызунов, viv1313r@mail.ru

Санкт-Петербургский университет ГПС МЧС России,
Санкт-Петербург, 196105, Российская Федерация

Аннотация

Актуальность. *Современные информационные системы функционируют в условиях перманентного действия гибридных дестабилизирующих факторов, природа которых – от целенаправленных кибератак до стохастических технических сбоев, саботажа, ухода ключевого персонала, введения санкций – часто априорно неизвестна. Существующие методы управления информационными системами в условиях дестабилизирующих факторов фрагментарны: они либо сосредоточены на узких технических аспектах, либо ограничены административными регламентами, не обеспечивая целостного охвата всех иерархических уровней системы.*

Цель. *Разработка универсальной архитектурной модели управления ONYX, представляющей информационную систему как вычислимое пространство состояний и обеспечивающей верифицируемую адаптацию к дестабилизирующим факторам произвольной природы для сохранения ее функционала.*

Методы. *Состояние информационной системы представлено как атрибутированный мультиграф. Для иерархических систем введен Постулат иерархической организованности с делением на уровни: обеспечивающий, персонала, аппаратного и программного обеспечения. Валидность и легитимность состояний определяется соответственно предикатами IsValid(s) и IsIntended(s) на основе логики первого порядка. Управление системой реализуется оператором R на базе верифицированных шаблонов.*

Результаты. *Разработана модель ONYX, представляющая информационную систему в виде мультиграфа с оператором управления. Доказаны теоремы о критерии существования решения (X-адаптивность), разрешимости восстановления оператором R, инвариантности безопасности и относительной полноты оператора. Выведены следствия: эффект «Конуса влияния» и принцип уровневой нейтрализации угроз.*

Научная новизна *заключается в универсальности формализма для иерархических и роевых систем, определении необходимых и достаточных условий разрешимости задачи восстановления, доказательстве следствий иерархии, обосновании кросс-уровневой восстанавливаемости и введении инварианта валидности автоматического управления.*

Теоретическая значимость: *создание математического аппарата для управления структурной динамикой сложных систем в условиях дестабилизирующих факторов произвольной природы. Практическая значимость – в формализации требований ИБ-стандартов (ISO/IEC 27001, ГОСТ Р ИСО/МЭК 270xx, NIST CSF) для построения систем SOAR нового поколения, объединяющих организационные и технические меры.*

Ключевые слова: *графовая модель, информационная безопасность, логика первого порядка, упреждающее управление, X-адаптивность, иерархическая архитектура, социотехнические системы*

Ссылка для цитирования: Грызунов В.В. Архитектурная модель ONYX для X-адаптивного управления информационной системой в условиях дестабилизирующих факторов произвольной природы // Труды учебных заведений связи. 2026. Т. 12. № 3. С. 112–128. DOI:10.31854/1813-324X-2026-12-3-112-128. EDN:JIZXYX

Original research
https://doi.org/10.31854/1813-324X-2026-12-3-112-128
EDN:JIZXYX

Architectural Model ONYX for X-Adaptive Control of Information Systems Under Arbitrary Destabilizing Factors

Vitaliy V. Gryzunov, viv1313r@mail.ru

Saint-Petersburg University of the State Fire Service of the EMERCOM of Russia,
St. Petersburg, 196105, Russian Federation

Annotation

Background. *Modern information systems operate under the permanent influence of hybrid destabilizing factors whose nature — ranging from targeted cyberattacks to stochastic technical failures, sabotage, key personnel departure, and sanctions imposition — is often a priori unknown. Existing methods for controlling information systems under destabilizing factors are fragmented: they either focus on narrow technical aspects or are limited to administrative regulations, failing to provide holistic coverage of all hierarchical levels of the system.*

Objective. *The objective is to develop a generic architectural control model, ONYX, representing an information system as a computable state space and ensuring verifiable adaptation to destabilizing factors of arbitrary nature in order to preserve its functionality.*

Methods. *The state of an information system is represented as an attributed multigraph. For hierarchical systems, the Hierarchical Organization Postulate is introduced, decomposing the graph into the following levels: Management, Personnel, Hardware, and Software. The validity and intendedness of states are determined by the predicates* $IsValid(s)$ *and* $IsIntended(s)$*, respectively, based on first-order logic. System control is implemented by the operator R using a database of verified templates.*

Results. *The ONYX model has been developed, representing an information system as a multigraph with a control operator. Theorems have been proved on the solution existence criterion, namely X-adaptivity, on the solvability of recovery by the operator R, on safety invariance, and on the relative completeness of the operator. The following consequences have been derived: the "Cone of Influence" effect and the principle of layer-based threat neutralization.*

The scientific novelty *lies in the universality of the formalism for hierarchical and swarm systems, the definition of necessary and sufficient conditions for the solvability of the recovery task, the proof of the consequences of hierarchy, the substantiation of cross-level recoverability, and the introduction of a validity invariant for automatic control.*

Theoretical and Practical Significance. *The theoretical significance consists in creating a mathematical apparatus for controlling the structural dynamics of complex systems under destabilizing factors of arbitrary nature. The practical significance lies in formalizing the requirements of information security standards, including ISO/IEC 27001, GOST R ISO/IEC 270xx, and NIST CSF, for building next-generation SOAR systems that integrate organizational and technical protection measures.*

Keywords: *graph model, information security, first-order logic, preemptive control, X-adaptability, hierarchical architecture, socio-technical systems*

For citation: Gryzunov V.V. Architectural Model ONYX for X-Adaptive Control of Information Systems Under Arbitrary Destabilizing Factors. *Proceedings of Telecommunication Universities.* 2026;12(3):111–127. (in Russ.) DOI:10.31854/1813-324X-2026-12-3-112-128. EDN:JIZXYX

1. Введение

Современные информационные системы (ИС) функционируют в условиях постоянно возрастающего давления со стороны дестабилизирующих факторов (ДФ) различной природы: от целенаправленных кибератак и стохастических технических сбоев до организационных ошибок, дефицита финансирования и ухода ключевых специалистов. Во-первых, на программно-аппаратный уровень

идут кибератаки: за последний год в России общее число инцидентов информационной безопасности (ИБ) в 2024 г. выросло в 2,5 раза[1]. Мировые убытки от киберпреступности составили 9,22 трлн долларов в 2025 г. и достигнут 13,82 трлн к 2028 г.[2]. Во-вторых, макроэкономические и геополитические факторы искажают условия существования ИС: в 2022 г. 68,5 % российских компаний столкнулись с проблемами из-за ухода зарубежных вендоров[3]. В-третьих, сама сложность ИС является внутренним источником нестабильности: рост числа пользователей до 5,64 млрд и объема данных до 180 ZB к 2025 г. создает колоссальную нагрузку, при которой час простоя критической системы стоит свыше 1 млн долларов для 44 % организаций[4]. Возникают деструктивные факторы инфраструктурного генеза [1].

Существующие подходы к управлению ИС в условиях ДФ, как правило, фрагментарны: они либо фокусируются исключительно на программно-аппаратном уровне, например, методы машинного обучения для обнаружения аномалий или теоретико-игровые модели реагирования на атаки, либо ограничиваются процессным соответствием международным стандартам (ISO/IEC 27001, NIST CSF). Однако ни один из этих подходов не обеспечивает целостного охвата всех уровней ИС – от документов и бюджетов до квалификации персонала, аппаратуры и программного обеспечения. В результате даже эффективные тактические решения не устраняют первопричины инцидентов, лежащие на более высоких иерархических уровнях.

В условиях высокой неопределенности, когда ДФ не могут быть заранее перечислены, традиционное оптимальное управление неприменимо. На смену ему приходит адаптивное управление – гибкий, самонастраивающийся подход, способный реагировать на возмущения произвольной природы. Однако и здесь сохраняется разрыв: большинство моделей адаптации по-прежнему игнорируют организационно-техническую природу ИС и не формализуют взаимосвязь между ее уровнями.

Задача обеспечения штатного функционирования ИС в условиях ДФ является предметом активных исследований. Так, например, методы на основе машинного обучения способны эффективно обнаруживать и восстанавливать данные на тактическом уровне [2], однако не решают проблему противодействия ДФ в широком смысле, игнорируя первопричины уязвимостей на более высоких иерархических уровнях – аппаратном, обеспечивающем или персонала. Подходы, основанные на концепции доверия [3], изолируют узлы с вредоносным поведением, но оперируют качественными метриками и не позволяют выявить первопричину сбоя, лежащую на высших уровнях иерархии ИС.

Обучение с подкреплением [4] способно вырабатывать оптимальные стратегии защиты с помощью автономных агентов [5], однако их эффективность ограничена обучающими данными, что делает их уязвимыми перед лицом новых угроз. Теоретико-игровые модели [6] находят математически обоснованные стратегии распределения ресурсов, но требуют наличия модели атакующего и неэффективны в условиях неопределенности, когда дестабилизация вызвана сбоем, а не рациональной атакой. Модели на основе живучести [7] направлены на сохранение структурной целостности системы, в том числе иерархической [8], но не на поддержание производительности, что в первую очередь интересно пользователю.

Децентрализованные подходы [9] противодействуют отказам отдельных узлов, а механизмы стимулирования [10] позволяют координировать действия агентов, но сама их природа затрудняет реализацию глобальных, стратегических управляющих воздействий, таких как изменение регламентов или обучение персонала. Методы теории управления [11], устойчивых наблюдателей [12] и адаптивной идентификации [13] имеют математическую строгость, но обладают формальной моделью, адекватно описывающей лишь программно-аппаратный уровень.

Отдельно стоит отметить исследования в области киберфизических систем. Подходы, основанные на формальных моделях [14] и онтологиях [15], на метамоделировании [16], эффективны на этапе проектирования, но не предлагают механизмов для адаптивного управления в реальном времени. Работы, фокусирующиеся на классификации угроз [17] и уязвимостей [18], создают структурированную базу знаний, но являются описательными, а не управленческими. Даже глубокие архитектуры человеко-машинных систем [19], признающие роль человека, остаются на высоком уровне абстракции и не отвечают на вопрос, как разнородные факторы влияют на измеримый показатель эффективности системы.

Таким образом, проведенный анализ выявляет два ключевых разрыва: практически все подходы ограничиваются рассмотрением программно-аппаратного уровня, не в полной мере учитывая связь

[1]The number of cyber attacks in Russia and in the world // TAdviser. 2026. URL: https://tadviser.com/index.php/Article:The_number_of_cyber_attacks_in_Russia_and_in_the_world (Accessed 11.06.2026)

[2] URL: https://protectmyapp.com/cybersecurity/knowledge-base/cybercrime-statistics-key-stats-and-insights (Accessed 28.06.2026)

[3] Источник: Аналитика CNews, TAdviser, 2022–2023 гг.

[4]Global Server Hardware, Server OS Reliability Report 2022–2023. ITIC.

с уровнями обеспечения (документы, регламенты, финансы) и персонала, и опираются на предзаданные модели угроз, что снижает их эффективность в условиях неопределенности.

Настоящая работа направлена на устранение указанных разрывов. Целью исследования является разработка архитектурной модели адаптивной системы управления ИС, способной противодействовать ДФ произвольной природы и интегрировать все уровни ИС в единое вычислимое представление, т. е. формализованную структуру данных, допускающую алгоритмическую проверку валидности, расчет производительности и генерацию управляющих воздействий.

2. Иерархическая структура ИС

2.1. Формальная постановка задачи

В основе предлагаемого подхода лежит разделение на универсальное вычислительное ядро модели ONYX и предметную область, в которой функционирует ИС.

Формализм: состояние системы моделируется как атрибутированный мультиграф общего вида, что обеспечивает применимость подхода к топологиям произвольной сложности, включая mesh-сети, роевые структуры и др.

Предметное ограничение: для класса рассматриваемых в статье ИС вводится Постулат иерархической организованности (HOP, *аббр. от англ.* Hierarchical Organization Postulate). В любой момент времени t граф состояния $G(t)$ допускает декомпозицию на упорядоченное множество уровней:

$$\mathcal{L} = \{L_1, L_2, \dots, L_n\},$$

где:

– каждый уровень L_i представляет собой семантически однородную подсистему, например, обеспечивающую, персонала, аппаратную, программную природу;

– множества элементов уровней попарно дизъюнктны (это обеспечивается механизмом глобальной уникальной идентификации (ГУИ));

– иерархия реализуется через направленное влияние от вышестоящих уровней к нижестоящим.

Постановка задачи на исследование

Дано:

1) современная информационная система $\mathcal{I}$, функционирующая в условиях действия ДФ произвольной природы;

2) требования нормативных стандартов (ISO/IEC 27001, ГОСТ Р ИСО/МЭК 27035 и др.) к управлению соответствием и реагированию на инциденты ИБ;

3) отсутствие единой формализованной структуры, интегрирующей все уровни ИС в единое вычислимое представление.

Требуется разработать архитектурную основу модели адаптивного управления ИС, удовлетворяющую следующим условиям:

– формализация состояния ИС как единого атрибутированного мультиграфа, построенного на основе онтологии $\mathcal{O} = (\mathcal{L}, \Gamma, K, \mathcal{D})$, с попарно дизъюнктными множествами вершин и межуровневыми связями;

– определение предиката легитимности $\mathrm{IsIntended}(s)$, вычислимого за полиномиальное время, формально разделяющий состояния на штатные ($S_{intended}$), инцидентные, но допустимые ($S_{incident}$), недопустимые, т. е. физически невозможные ($S \setminus S_{valid}$);

– обеспечение масштабируемости архитектуры и поддержки X-адаптивного управления как структурного свойства.

2.2. Формальное определение пространства состояний

Модель ONYX включает онтологию $\mathcal{O}$ и пространство состояний S:

$$\mathcal{O} = (\mathcal{L}, \Gamma, K, Dom),$$

где $\mathcal{L}$ – конечное множество уровней иерархии, например $\{Ens, P, Hard, Soft\}$; Γ – конечное множество типов связей, например, {администрирует, размещается_на, финансирует}; K – множество имен атрибутов; Dom – отображение, сопоставляющее каждому $k \in K$ конечный домен значений Dom_k.

Онтология $\mathcal{O}$ явно задается при проектировании системы и может быть расширена без изменения ядра архитектуры ONYX. В рамках настоящей работы с сохранением общности рассматривается четырехуровневая реализация онтологии $\mathcal{L}$, соответствующая обеспечивающему уровню ($\ell_1 = Ens$), уровню персонала / кадровому уровню ($\ell_2 = P$), аппаратному уровню ($\ell_3 = Hard$) и программному ($\ell_4 = Soft$):

$$\mathcal{L} = \{Ens, P, Hard, Soft\}.$$

Выбор четырехуровневой структуры ONYX обоснован практикой управления ИБ и соответствует доменам ведущих стандартов: обеспечивающий уровень отражает требования к лидерству и ресурсам[5]; уровень персонала – к компетентности[6] и культуре управления[7]; аппаратный и программный

[5] ISO/IEC 27001:2022 Clause 5, 7.1, ГОСТ Р ИСО/МЭК 27003-2012 Clause 7.1,

[6] ГОСТ Р ИСО/МЭК 27021-2021, ISO/IEC 27001:2022 Clause 7.2

[7] ISO/IEC 27001:2022 Annex A.8 Asset Management, COBIT

уровни – к управлению активами и технологическими средствами[8].

Важно подчеркнуть, что архитектура ONYX не привязана жестко к этим четырем уровням. Онтология $\mathcal{L}$ является расширяемым компонентом модели, что позволяет вводить новые уровни, например, «виртуальная структура», или агрегировать существующие без изменения ядра архитектуры путем переопределения $\mathcal{L}$. Пространство состояний S ИС определяется как множество атрибутированных мультиграфов, построенных на основе фиксированной онтологии уровней.

Пусть Γ – конечное множество типов связей, Σ – конечный алфавит для построения глобальных уникальных идентификаторов (ГУИ), а Σ^* – множество всех конечных строк над Σ (включая пустую строку).

Тогда состояние ИС в момент времени $t \in T \subseteq \mathbb{N}$ – это тройка:

$$s(t) = (V(t), E(t), \text{prop}),$$

где $V(t)$ – множество вершин:

$$V(t) \subseteq \bigcup_{\ell \in \mathcal{L}} (\{\ell\} \times \Sigma^*),$$

благодаря представлению вершин в виде упорядоченных пар $v = (\ell, \sigma)$, где $\ell \in \mathcal{L}$ – метка уровня, а $\sigma \in \Sigma^*$ – локальный идентификатор, принадлежность к разным уровням становится неотъемлемым свойством элемента; это обеспечивает онтологическую дизъюнктность уровней на уровне формальной модели; на практике уникальность всех элементов, в том числе внутри уровня, гарантируется ГУИ, согласно которому полный идентификатор строится как конкатенация префикса категории и локального имени (*P*:*adm_db*_01, *Soft*:*pgsql*_v14); $E(t)$ – множество ребер определяется как подмножество декартова произведения вершин:

$$E(t) \subseteq V(t) \times V(t),$$

каждое ребро $e = (u, v) \in E(t)$ соединяет вершину-источник u с вершиной-целью v; типизация связей реализуется через функцию атрибутов prop; для каждого ребра *e* определен обязательный атрибут '*type*' со значением из фиксированного словаря типов Γ:

$$\forall e \in E(t): \text{prop}(e, \text{'}type\text{'}) \in \Gamma,$$

на основе значений этого атрибута и принадлежности вершин уровням множество $E(t)$ семантически разбивается на внутриуровневые ребра $E_{internal}(t)$, связывающие вершины одного уровня ($\ell_u = \ell_v$), и межуровневые $E_{cross}(t)$, связывающие вершины разных уровней ($\ell_u \neq \ell_v$); функция атрибутов prop является частичной функцией, определенной на парах «элемент–имя атрибута» и принимающей значения из конечных доменов:

$$\text{prop}: (V(t) \cup E(t)) \times K \rightharpoonup \bigcup_{k \in K} Dom_k,$$

где K – множество имен атрибутов из онтологии $\mathcal{O}$; Dom_k – конечный домен значений атрибута k, например, $Dom_{competence}$ = {1, 2, 3, 4, 5}, Dom_{load} = {0, 1, ..., 100}, Dom_{status} = {«работает», «остановлен», «сбой»}, $Dom_{finance}$ = {1, … , M}.

Следовательно, полное пространство состояний:

$$S = \{s(t) = (V(t), E(t), \text{prop}) \mid t \in T\}.$$

Приведем содержательные характеристики каждого уровня ИС.

2.3. Обеспечивающий уровень (*Ens*)

Ens – источник законодательных требований и финансирования для всех нижележащих уровней.

Формально: вершины уровня *Ens* – элементы множества $V_{Ens}(t) = \{v = (Ens, \sigma) \mid v \in V(t)\} \subset V(t)$.

Пример: «Политика ИБ v1.2» ↔ вершина $v = (Ens, policy\_ib\_v1.2) \in V_{Ens}(t)$.

Типичные вершины (практические примеры): «Бюджет ИБ» ↔ $v_{budget} = (Ens, budget\_ib\_2025)$, «Политика ФСТЭК №117» ↔ $v_{fsk} = (Ens, fsk\_117)$.

Типичные атрибуты (prop):

$$\text{prop}(v_{budget}, value) = 5\,000\,000,$$
$$\text{prop}(v_{budget}, currency) = \text{«}RUB\text{»},$$
$$\text{prop}(v_{fsk}, type) = \text{«документ»},$$
$$\text{prop}(v_{fsk}, name) = \text{«ФСТЭК №117»},$$
$$\text{prop}(v_{fsk}, effective\_date) = \text{«2026-03-01»}.$$

Для удобства представления совокупности атрибутов вершины v далее используется сокращенная запись:

$$\text{prop}(v) = \{k_1 = v_1, k_2 = v_2, \dots\},$$

которая обозначает набор значений функции $\text{prop}(v, k)$ для всех $k \in K$, определенных на v.

Примеры:

$$\text{prop}(v_{budget}) = \{type = \text{«финансы»}, value = 5\,000\,000, currency = \text{«}RUB\text{»}\},$$
$$\text{prop}(v_{fsk}) = \{type = \text{«документ»}, name = \text{«ФСТЭК №117»}, effective\_date = \text{«2026-03-01»}\}.$$

2.4. Уровень персонала (*P*)

Уровень персонала (кадровый уровень) включает кадры, их компетенции и структура, человеческий интерфейс реализации замысла.

[8] ISO/IEC 27001 A.7–A.8, COBIT Technology

Формально: вершины уровня P – элементы множества $V_P(t) = \{v = (P, \sigma) \mid v \in V(t)\}$.

Типичные вершины: «Администратор СУБД» ↔ $v_{adm} = (P, adm\_db\_01) \in V_P(t)$, «Роль: Специалист по SIEM» ↔ $v_{role} = (P, role\_siem)$.

Типичные атрибуты:

$$\text{prop}(v_{adm}) = \{type = \text{«сотрудник»}, competence = 4, load = 0{,}7\},$$

$$\text{prop}(v_{role}) = \{type = \text{«роль»}, required\_competence = 4, criticality = \text{«высокая»}\}.$$

2.5. Аппаратный уровень (*Hard*, АО)

Аппаратный уровень (АО) содержит физические ресурсы: серверы, сети, хранилища, среду для исполнения программного обеспечения (ПО).

Формально: вершины уровня *Hard* – элементы $V_{Hard}(t) = \{v = (Hard, \sigma) \mid v \in V(t)\}$.

Типичные вершины:

«Сервер СУБД» ↔ $v_{srv} = (Hard, srv\_db\_01) \in V_{Hard}(t)$,

«Канал связи 10 Гбит/с» ↔ $v_{link} = (Hard, link\_10g)$.

Типичные атрибуты:

$$\text{prop}(v_{srv}) = \{type = \text{«сервер»}, cpu\_cores = 32, memory\_gb = 256, status = \text{«работает»}\},$$

$$\text{prop}(v_{link}) = \{type = \text{«канал»}, bandwidth\_mbps = 10000\}.$$

2.6. Программный уровень (*Soft*, ПО) описывает все программные компоненты и их зависимости, результат работы ИС.

Формально: вершины уровня *Soft* – элементы $V_{Soft}(t) = \{v = (Soft, \sigma) \mid v \in V(t)\}$.

Типичные вершины:

«СУБД *PostgreSQL*» ↔ $v_{db} = (Soft, pgsql\_v14) \in V_{Soft}(t)$,

«*SIEM*-сервер» ↔ $v_{siem} = (Soft, siem\_v2.1)$.

Типичные атрибуты:

$$\text{prop}(v\_db) = \{type = \text{«сервис»}, name = \text{«}PostgreSQL\text{»}, version = \text{«14,5»}, status = \text{«работает»}\},$$

$$\text{prop}(v\_siem) = \{type = \text{«сервис»}, resource\_demand = \{cpu: 5, storage: 20\}\}.$$

2.7. Иерархия как направленное влияние

Иерархия в ONYX проявляется не в изолированности уровней, а в направленности межуровневых связей и семантике правил валидности.

Ребро типа «администрирует»:

$$\text{от } v_{adm} \in V_P(t) \text{ к } v_{db} \in V_{Soft}(t).$$

Ребро типа «размещает»:

$$\text{от } v_{srv} \in V_{Hard}(t) \text{ к } v_{db} \in V_{Soft}(t).$$

Ребро типа «финансирует»:

$$\text{от } v_{budget} \in V_{Ens}(t) \text{ к } v_{srv} \in V_{Hard}(t).$$

Таким образом реализуется цепочка замысла:

Замысел (*Ens*) → Исполнитель (*P*) → Среда (*Hard*) → Результат (*Soft*).

Нарушение на любом этапе – признак отклонения от целевого замысла, даже если система продолжает функционировать. Отслеживание соответствия замыслу и позволяет ONYX обеспечивать упреждающее управление.

Чтобы отличить все теоретически возможные состояния ИС от физически допустимых, вводится предикат валидности:

$$\text{IsValid}: s \to \{true, false\},$$

определяющий подмножество допустимых состояний: $S_{valid} = \{s \in S \mid \text{IsValid}(s) = true\}$.

3. Предикаты валидности IsValid(s) и легитимности IsIntended(s) на основе логики первого порядка

Предикат валидности $\text{IsValid}(s)$ формализует условия целостности и физической реализуемости модели: состояние s валидно, если граф синтаксически корректен и соблюдены зависимости, например, каждый программный компонент имеет аппаратную платформу. Нарушение $\text{IsValid}(s)$ делает адаптацию некорректной.

Предикат легитимности $\text{IsIntended}(s)$ формализует штатные (легитимные) состояния системы.

IsValid и IsIntended задаются как конъюнкция правил в логике первого порядка (FOL).

3.1. Универсум и базовые предикаты

Универсум рассуждений – это конечное множество всех элементов модели:

$$M_{ONYX} = V(t) \cup E(t).$$

Все переменные в формулах явно типизируются: $v, u, \ldots$ обозначают вершины ($v \in V(t)$), $e, f, \ldots$ обозначают ребра ($e \in E(t)$). Соответственно, кванторы в правилах всегда имеют подразумеваемый тип: $\forall v$ означает $\forall v \in V(t)$, $\exists e$ означает $\exists e \in E(t)$. Это гарантирует, что предикаты вроде $\text{Level}(v, P)$ или $\text{Source}(e, v)$ применяются только к аргументам корректных типов, что исключает синтаксически некорректные формулы.

Предикаты и функции модели:

– $\text{Level}(x, \ell)$ – предикат, определенный только для вершин $x \in V(t)$, истинный тогда и только тогда, когда вершина x принадлежит уровню $\ell \in \mathcal{L}$;

– $\text{Source}(e, v)$, $\text{Target}(e, u)$ связывают ребро $e \in E(t)$ с его исходной и целевой вершинами $v, u \in V(t)$.

Функция атрибутов. Для любого элемента $x \in V(t) \cup E(t)$ и любого ключа атрибута $k \in K$ значение атрибута определяется частичной функцией $\text{prop}(x, k) \in Dom_k$. Функция является единственным механизмом доступа к атрибутивной информации модели.

Различие между внутриуровневыми и межуровневыми связями выражается через предикат Level: ребро *e* является межуровневым, если:

$$\exists v, u, \ell_1, \ell_2(\text{Source}(e, v) \wedge \text{Target}(e, u) \wedge \\ \wedge \text{Level}(v, \ell_1) \wedge \text{Level}(u, \ell_2) \wedge \ell_1 \neq \ell_2).$$

Формальное определение. Предикат валидности определяется как конечная конъюнкция замкнутых формул из разрешимого фрагмента логики первого порядка, основанного на конъюнктивных запросах (CQ, *аббр. от англ.* Conjunctive Queries) и включающего Tuple-Generating Dependencies (TGD) и Equality-Generating Dependencies (EGD):

$$\text{IsValid}(s) \equiv \bigwedge_{i=1}^{n} \text{Rule}_i\,(s),$$

где каждое $\text{Rule}_i(s)$ является замкнутой формулой вида:

$$\forall \vec{x}\,\big(\phi(\vec{x}) \rightarrow \exists \vec{y}\,\psi(\vec{x}, \vec{y})\big), \text{класс TGD},$$

$$\forall \vec{x}\,(\phi(\vec{x}) \rightarrow x_1 = x_2), \text{класс EGD},$$

где ϕ и ψ – *CQ* над предикатами модели.

Предикат IsIntended(*s*) определяется аналогично.

Проверка предикатов IsValid(*s*) и IsIntended(*s*) обладает сложностью *LOGSPACE* по данным (*Data Complexity*), то есть потребление памяти растет полилогарифмически при фиксированном наборе правил и растущем размере графа системы [20]. Это гарантирует масштабируемость метода для промышленных ИС высокой размерности. В общем случае (*Combined Complexity*) сложность определяется выразительной силой выбранного фрагмента логики, для рассматриваемого класса TGD без циклов зависимости она остается полиномиальной.

Примечание. Расширение логического базиса. Для работы с параметрическими атрибутами (пропускная способность, загрузка *CPU*) множество допустимых атомов в формулах $\text{Rule}_i(s)$ дополняется встроенными предикатами сравнения (*Built-in Predicates*) $\bowtie \in \{<, \leq, >, \geq, =\}$ над доменами значений атрибутов. Поскольку согласно п. 2.2 все домены Dom_k конечны, введение арифметических сравнений не выводит задачу проверки валидности за пределы класса сложности *LOGSPACE* (*Data Complexity*).

Примеры правил. Пусть в онтологии задано правило зависимости: «Любой компонент типа *service* на уровне *Soft* с атрибутом *criticality* = *high* должен управляться субъектом на уровне *P*, обладающим *competence* ≥ 4».

Формально это выражается как схема:

$$\forall v(\text{Level}(v, \ell_1) \wedge \text{Type}(v, t) \wedge \text{prop}(v, k) = v_k \rightarrow \\ \rightarrow \exists u \exists e(\text{Level}(u, \ell_2) \wedge \text{EdgeType}(e, \tau) \wedge \\ \wedge \text{Source}(e, u) \wedge \text{Target}(e, v) \wedge \text{prop}(u, k') \bowtie \theta)),$$

где $\ell_1, \ell_2, t, \tau, k, v_k, k', \bowtie, \theta$ – параметры правила, заданные не коде, а в метаописании правила.

После подстановки правило выглядит так:

$$\forall v(\text{Level}(v, \textit{Soft}) \wedge \text{Type}(v, \textit{service}) \wedge \\ \wedge \text{prop}(v, \textit{criticality}) = \textit{high}) \rightarrow \exists u \exists e \exists c(\text{Level}(u, P) \wedge \\ \wedge \text{EdgeType}(e, \text{администрирует}) \wedge \text{Source}(e, u) \wedge \\ \wedge \text{Target}(e, v) \wedge \text{prop}(u, \textit{competence}) = c \wedge (c = 4 \vee c = 5))).$$

Пояснение подстановки:
$\ell_1 = \textit{Soft}$ – программный уровень;
$\ell_2 = P$ – уровень персонала;
τ = администрирует – тип межуровневой связи (значение prop(*e*, '*type*'));
t = *service* – тип вершины (значение prop(*v*, '*type*'));
k = *criticality*, v_k = high – ключ и значение атрибута, определяемые равенством $\text{prop}(v, k) = v_k$; неравенство в правой части $\text{prop}(u, k') \geq \theta$ проверяет, что значение компетенции администратора (атрибут $k' = \textit{competence}$) не ниже требуемого порога $\theta = 4$.

Таким образом, правила становятся шаблонами, применимыми к любой паре уровней и типов, описанных в онтологии.

В модели ONYX важно различать синтаксис формул и топологию графа. Прямая запись вида Source(*v*, *v*) невозможна, так как это является ошибкой типов: первый аргумент предиката обязан быть ребром, а не вершиной. Однако структурные петли допустимы: они моделируются через существование ребра *e*, которое одновременно выходит из вершины и входит в нее, то есть истинны оба предиката: Source(*e*, *v*) и Target(*e*, *v*). Такие петли легитимны, если тип связи *e* семантически допускает рефлексию, например, «самодиагностика» или «внутренний цикл.

3.2. Декомпозиция пространства валидных состояний

Для классификации ситуаций внутри множества валидных (физически реализуемых) состояний S_{valid} водятся дополнительные целевые предикаты:

– $P_{\text{realizable}}(s) \equiv \text{IsValid}(s)$ проверяет физическую и логическую возможность существования состояния, например, «ПО не может существовать без АО»;

– $P_{\text{compliant}}(s)$ – соответствие политикам и регламентам;

– $P_{\text{target}}(s)$ – соответствие целевому архитектурному замыслу, например, наличие резерва для критического сервиса.

Предикаты связаны между собой так:

$$IsIntended(s) = \text{IsValid}(s) \wedge P_{\text{compliant}}(s) \wedge P_{\text{target}}(s).$$

На основе предикатов вводится разбиение:

$$S_{valid} = S_{intended} \cup S_{incident}, S_{intended} \cap S_{incident} = \emptyset,$$

где $S_{intended}$ – штатные (легитимные) состояния, система физически возможна, безопасна и соответствует проекту:

$$S_{intended} = \{ s \in S_{valid} \mid P_{\text{compliant}}(s) \wedge P_{\text{target}}(s) \};$$

($S_{incident}$) – инцидентные состояния, система функционирует, но нарушены политики или архитектурные требования:

$$S_{incident} = \{s \in S_{valid} \mid \neg\big(P_{\text{compliant}}(s) \wedge P_{\text{target}}(s)\big)\}.$$

Таким образом, ИБ в модели ONYX – это не обязательно нарушение доступности, целостности или конфиденциальности в классическом понимании, а любое событие, в результате которого ИС переходит из подмножества штатных состояний $S_{intended}$ в подмножество инцидентных состояний $S_{incident}$ (критический сервис не имеет назначенного резервного администратора; политика безопасности требует двухфакторной аутентификации, но она отключена для подгруппы пользователей без формального исключения; документация по эксплуатации устарела, хотя система функционирует).

Поскольку $S_{valid} = S_{intended} \cup S_{incident}$, и $\text{IsValid}(s) \equiv P_{\text{realizable}}(s)$, то нарушение $P_{\text{realizable}}$ означает выход за пределы моделируемой реальности (невалидное состояние), тогда как нарушение $P_{\text{compliant}}$ или P_{target} является триггером для адаптивного управления.

4. Оператор управления *R*

4.1. Дестабилизирующие факторы (ДФ) и их влияние на модель

Пусть $\mathcal{D}$ – множество всех возможных ДФ. Каждому фактору $d \in \mathcal{D}$ сопоставляется частичный оператор трансформации состояния $F_d: S \rightharpoonup S$, переводящий систему из состояния s в возмущенное состояние $s' = F_d(s)$.

Множество $\mathcal{D}$ классифицируется по типу воздействия на граф $G = (V, E, \text{prop})$:

Структурные факторы ($\mathcal{D}_{struct}$): изменяют топологию, то есть $(V', E') \neq (V, E)$. *Примеры*: физический выход из строя сервера (удаление вершины $v \in V_{Hard}$), разрыв канала связи (удаление ребра $e \in E_{cross}$), увольнение сотрудника (удаление $v \in V_P$).

Параметрические факторы ($\mathcal{D}_{par}$): сохраняют топологию, но изменяют атрибуты: $(V', E') = (V, E)$ и $\exists x, k: \text{prop}'(x, k) \neq \text{prop}(x, k)$. *Примеры:* снижение пропускной способности канала (атрибут *bandwidth*), устаревание версии ПО (атрибут *actual_version*), снижение эффективности сотрудника (атрибут *competence*).

Смешанные факторы: затрагивают и структуру, и параметры.

По природе ДФ: $\mathcal{D} = \mathcal{D}_{det} \cup \mathcal{D}_{st} \cup \mathcal{D}_{unc}$, где $\mathcal{D}_{det}$ (детерминированные) – плановые изменения с известным результатом; $\mathcal{D}_{st}$ (стохастические) – сбои / отказы с известным распределением вероятностей; $\mathcal{D}_{unc}$ (неопределенные) – воздействия, модель которых априорно неизвестна или неприменима (новые уязвимости, целенаправленные атаки [21]).

Определение. $\mathcal{X}$-возмущением в модели – фактор $d \in \mathcal{D}$, действие которого сохраняет физическую реализуемость состояния, но увеличивает отклонение от целевого архитектурного замысла относительно некоторой метрики μ:

$$X = \{d \in \mathcal{D} | \exists s \in S_{valid}: F_d(s) \in S_{incident} \wedge \mu(F_d(s)) > \mu(s)\}.$$

Этот ДФ нарушает целевой замысел, но не выводит систему за пределы S_{valid}, следовательно, к X-возмущению применимы методы адаптивного управления [22]. Оператор R реагирует непосредственно на отклонение состояния $s \to s'$, без идентификации фактора d. Модели безразлично, была ли связь разорвана экскаватором или хакером – важно, что ребро исчезло или его параметр *bandwidth* упал ниже порога.

4.2. Определение оператора управления *R*

Оператор управления R – ключевой компонент ONYX, задающий динамику переходов между состояниями системы и обеспечивающий адаптивную самонастройку системы. Его задача – генерировать структурно корректные изменения, удерживающие систему в S_{valid} и возвращающие ее в $S_{intended}$. Вместо вычислительно неразрешимого перебора всех трансформаций R синтезирует управляющие воздействия на основе конечной базы верифицированных шаблонов $\mathcal{P}$.

Определение. База шаблонов $\mathcal{P} = \{\pi_1, \dots, \pi_m\}$ – это конечное множество параметризованных правил трансформации графа, где каждый шаблон π_k определяет:

– предусловие (Precondition): структурный паттерн графа, соответствующий определенному типу инцидента, например, «Узел типа Сервис не имеет связи с узлом типа Сотрудник»;

– действие (Action): последовательность элементарных операций π, устраняющих этот дефицит, например, «Создать ребро, связывающее Сервис с Сотрудником из пула резерва».

Набор изменений π определяется как упорядоченная последовательность элементарных операций из конечного множества Ω_{op}:

$$\pi = \langle \omega_1, \omega_2, \dots, \omega_k \rangle, \text{ где } \omega_i \in \Omega_{op}.$$

Множество Ω_{op} включает следующие операции:
add_vertex(id, $attrs$) добавляет новую вершину с идентификатором id и атрибутами $attrs$;
del_vertex(id) удаляет вершину id и все инцидентные ей ребра;

add_edge(u, v, *type*) добавляет ориентированное ребро от u к v с типом связи *type*;
del_edge(u, v, *type*) удаляет ребро указанного типа между u и v;
update_attr(*id*, *key*, *value*) обновляет значение атрибута *key* у элемента *id* на значение value.

Обозначим через ΔS множество всех конечных последовательностей операций из Ω_{op}:

$$\Delta S \stackrel{\text{def}}{=} \Omega_{op}^{*}.$$

Элемент $\pi \in \Delta S$ называется траекторией управления.

Функция Apply(s, π) последовательно применяет операции ω_i к графу состояния s. Каждая операция считается применимой, только если выполнены ее предусловия, например, удаляемая вершина существует, добавляемое ребро отсутствует и т. п. Если предусловие нарушено, последовательность π считается неприменимой.

Множество ΔS, упоминаемое в алгоритме ниже, – это множество всех конечных последовательностей операций из Ω_{op}, включая пустую, применимых хотя бы к одному состоянию $s \in S$.

Алгоритм работы оператора R представляет из себя определенную последовательность шагов.

Шаг 1. Диагностика: выявление подграфа $G_{defect} \subset G$, нарушающего условия соответствия замыслу или валидности.

Шаг 2. Поиск шаблонов (*Matching*): из базы $\mathcal{P}$ выбираются шаблоны, чьи предусловия соответствуют G_{defect}.

Шаг 3. Инстанцирование (*Instantiation*): для каждого подходящего шаблона генерируется конкретный набор изменений $\pi_i \in \Delta S$, где абстрактные параметры заменяются на идентификаторы реальных узлов, например, выбирается конкретный сотрудник «Иванов» с подходящими атрибутами.

Шаг 4. Верификация, для каждого сгенерированного набора изменений π_i выполняются следующие проверки:

– применимость: все операции из π_i могут быть выполнены над текущим состоянием s, например, удаляемая вершина существует, добавляемое ребро отсутствует;

– валидность результата: новое состояние $s' = \text{Apply}(s, \pi_i)$ удовлетворяет предикату $\text{IsValid}(s') = true$;

– проверяется условие прогресса: новое состояние $s' = \text{Apply}(s, \pi_i)$ должно быть строго ближе к целевому подмножеству $S_{intended}$, чем исходное s согласно некоторой метрики μ.

Например, суммарное число правил политик $P_{compliant}$ и условий целевого замысла P_{target}, нарушаемых в состоянии s:

$$\mu(s) = |\{r \in P_{target} \cup P_{compliant} \mid \neg \text{Rule}_r(s)\}|.$$

Если подходящий шаблон не найден, или ни один из кандидатов не прошел верификацию, например, нет свободных сотрудников, оператор возвращает исходное состояние s и генерирует сигнал эскалации для вмешательства человека-оператора. Применяется любой π_i, для которого $\mu(s') < \mu(s)$, например, первый кандидат. Это гарантирует отсутствие зацикливания при итеративном применении оператора R.

Применение: выбирается первый валидный набор π_i, и состояние системы обновляется: $s \leftarrow s'$.

Ключевым требованием к автоматизированной системе управления ИС является гарантия того, что управляющее воздействие не приведет к деградации системы или нарушению политик безопасности. В модели ONYX это обеспечивается конструктивно.

Утверждение 1. Инвариантность валидности (*от англ.* Safety Invariant)

Пусть $\mathcal{P}$ – база корректных шаблонов, такая что:

$$\forall \pi \in \mathcal{P}: \text{IsValid}(s) \wedge \text{Pre}(\pi, s) \Longrightarrow \text{IsValid}(\text{Apply}(s, \pi)),$$

где $\text{Pre}(\pi, s)$ – предикат, истинный в том случае, если к состоянию s применимы предусловия шаблона π (см. алгоритм поиска шаблонов в п. 4.2).

Тогда оператор R сохраняет инвариант валидности:

$$s \in S_{valid} \Longrightarrow R(s) \in S_{valid}.$$

Доказательство. Алгоритм R конструктивно ограничивает пространство выбора ΔS только подмножеством тех трансформаций из $\mathcal{P}$, пост-условия которых удовлетворяют предикату IsValid.

1) Множество кандидатов $\mathcal{P}$ конечно (по условию).

2) Применение каждого шаблона $\pi \in \mathcal{P}$ дает конкретное состояние:

$$s' = \text{Apply}(s, \pi),$$

3) Предикат IsValid разрешим (в модели ONYX проверяется за полиномиальное время), поэтому алгоритм R всегда завершится одним из результатов:

– $s' \in S_{valid}$ (если найден подходящий шаблон);

– s (если подходящих шаблонов нет).

Таким образом, $\forall s \in S_{valid}$: $R(s) \in S_{valid}$, то есть S_{valid} замкнуто относительно R, что гарантирует структурную корректность и внутреннюю согласованность всех управляющих воздействий.

Утверждение 2. $\mathcal{P}$-полнота восстановления (*от англ.* Relative Completeness)

Оператор R является полным относительно базы знаний $\mathcal{P}$: если существует траектория восстановления, представимая как композиция шаблонов из $\mathcal{P}$, то R найдет ее или эквивалентную по метрике μ за конечное число шагов.

Обоснование. Алгоритм R осуществляет полный перебор всех шаблонов из конечной базы P, их инстанцирование и верификацию. Если решение существует и охвачено базой знаний, оно будет найдено и применено. Таким образом, достижимость целевого состояния обеспечивается при достаточности базы шаблонов, что соответствует прагматическому подходу к автоматизации управления.

4.3. Концепция X-адаптивности

Модель ONYX реализует упреждающее управление.

Определение. Упреждающее управление – это режим функционирования системы, при котором оператор R инициирует корректирующие воздействия для любого валидного инцидентного состояния $s \in S_{incident} \subset S_{valid}$, характеризуемого ненулевым отклонением от целевого архитектурного замысла относительно некоторой метрики μ, то есть $\mu(s) > 0$.

Пример: отсутствие резервного администратора для критического сервиса не нарушает $P_{\text{compliant}}$, но нарушает P_{target}; состояние переходит в $S_{incident}$, что запускает R для назначения нового администратора до наступления критического сбоя; метрика μ – число нарушенных предикатов.

Переход в состояние $s \notin S_{valid}$ означает выход за пределы штатного режима функционирования модели ONYX, то есть характеризует ситуацию, когда упреждающее управление не отработало, и требует аварийного вмешательства, не охватываемого оператором R.

Традиционные методы управления разделяют робастность как устойчивость системы к малым параметрическим возмущениям при фиксированной структуре и структурную адаптацию как изменение топологии по заранее заданным правилам. Однако в условиях кибервойны и гибридных угроз ДФ из множества X может одновременно нарушать параметры и структуру системы, причем его природа заранее неизвестна. В отличие от классических подходов, X-адаптивность обеспечивает восстановление при таких неопределенных структурно-параметрических воздействиях.

Определение. Граф переходов состояний

Пусть Ω_{op} – множество элементарных операций системы.

Определим ориентированный граф:

$$\mathcal{G} = (S_{valid}, \mathcal{E}),$$

где S_{valid} – множество вершин (допустимых состояний); $\mathcal{E} \subseteq S_{valid} \times S_{valid}$ – множество дуг, таких что:

$$(s_i, s_j) \in \mathcal{E} \Leftrightarrow \exists \omega \in \Omega_{op}: s_j = \text{Apply}(s_i, \omega).$$

Определение. X-адаптивность

Система называется X-адаптивной, если для любого инцидентного состояния $s \in S_{incident}$ в графе $\mathcal{G}$ существует путь в целевое множество $S_{intended}$:

$$\forall s \in S_{incident}, \exists s^* \in S_{intended}, \exists \pi \in \Omega_{op}^*: \text{Apply}(s, \pi) = s^*,$$

где $\pi = \langle \omega_1, \dots, \omega_k \rangle$ – это траектория восстановления, то есть последовательность операций из Ω_{op}.

Примечание. Допустимые трансформации Ω_{op}^* включают и структурные (изменение топологии), и параметрические (изменение атрибутов) операции.

Физический смысл: X-адаптивность – это свойство архитектуры (пространства состояний), гарантирующее, что для любого инцидента существует хотя бы один допустимый путь восстановления; это свойство не зависит от полноты базы шаблонов $\mathcal{P}$ или природы ДФ.

Критерий X-адаптивности. Для формулировки критерия введем понятие Области притяжения (*от англ.* Attraction Basin):

$$\mathcal{R}^{-1}(S_{intended}) =$$
$$= \{s \in S_{valid} \mid \exists s^* \in S_{intended}: Reachable_{\mathcal{G}}(s, s^*)\}.$$

Теорема 1. О критерии существования решения (необходимое и достаточное условие)

ИС является X-адаптивной тогда и только тогда, когда множество инцидентных состояний является подмножеством области притяжения целевого множества:

$$S_{incident} \subseteq \mathcal{R}^{-1}(S_{intended}).$$

Доказательство:

– необходимость ($\Rightarrow$): по определению X-адаптивности для любого $s \in S_{incident}$ существует путь в $S_{intended}$; следовательно, каждый такой s обязан принадлежать множеству $\mathcal{R}^{-1}(S_{intended})$; если бы $s \in S_{incident}$, где s такое, что $s \notin \mathcal{R}^{-1}(S_{intended})$, то из него было бы невозможно восстановить систему, что противоречит определению;

– достаточность ($\Leftarrow$): в случае если $S_{incident} \subseteq \mathcal{R}^{-1}(S_{intended})$, то по построению множества $\mathcal{R}^{-1}$ для каждого инцидента гарантированно существует валидная траектория восстановления.

Теорема 1 определяет топологические требования к пространству состояний, допускающие последовательное восстановление.

Требование 1. Локальное требование (отсутствие тупиков / Deadlock Freedom): в графе $\mathcal{G}$ не должно быть состояний $s \in S_{incident}$, из которых не исходит ни одной дуги допустимого перехода. Физически это означает, что в любой ситуации, даже если целевой замысел нарушен, у системы должна оставаться возможность совершить валидный переход ($s \to s'$), где $s' \in S_{valid}$. При этом новое состояние s' может оставаться инцидентным, если полное восстановление за один шаг невозможно.

Требование 2. Глобальное требование (Отсутствие циклов / Livelock Freedom): последовательность локальных переходов должна формировать нециклическую траекторию, ведущую к выходу из множества $S_{incident}$ в $S_{intended}$. Недопустима ситуация, когда система бесконечно блуждает внутри множества инцидентных состояний (меняет структуру или параметры, оставаясь в $S_{incident}$), не имея возможности достичь целевого множества.

Связь *X*-адаптивности и реализации управления. Теорема 1 характеризует структурную возможность восстановления в графе $\mathcal{G}$. Но оператор R оперирует не всем графом допустимых переходов, а ограниченным набором знаний – базой шаблонов $\mathcal{P}$.

Определение $\mathcal{P}$-покрытия. База шаблонов $\mathcal{P}$ называется полной относительно $S_{incident}$, если любой теоретически существующий путь восстановления π может быть получен путем инстанцирования некоторого шаблона из $\mathcal{P}$.

Теорема 2. Об условии разрешимости восстановления

Задача автоматического восстановления разрешима оператором R тогда и только тогда, когда выполняются два условия:

1) система *X*-адаптивна (структурное условие): $S_{incident} \subseteq \mathcal{R}^{-1}(S_{intended})$;

2) база шаблонов $\mathcal{P}$ полна относительно $S_{incident}$ (информационное условие).

Доказательство:

– достаточность ($\Leftarrow$): пусть выполнены условия 1 и 2; докажем, что оператор R успешно переведет систему в $S_{intended}$;

– необходимость ($\Rightarrow$); пусть оператор R гарантированно решает задачу восстановления для любого $s \in S_{incident}$; докажем, что условия 1 и 2 выполняются.

Рассмотрим произвольное инцидентное состояние $s_0 \in S_{incident}$. Так как выполнено условие *X*-адаптивности (1), граф переходов $\mathcal{G}$ содержит хотя бы один путь $\pi_{exist} \in \Omega^*_{op}$, ведущий из s_0 в некоторое целевое состояние $s^* \in S_{intended}$, такой, что все промежуточные состояния валидны. Так как выполнено условие Полноты $\mathcal{P}$ (2), для существующего пути π_{exist} в базе знаний найдется шаблон $p \in \mathcal{P}$, инстанцирование которого порождает траекторию π_R, эквивалентную π_{exist}, или саму π_{exist}.

Согласно алгоритму работы оператора R (раздел 4.2), на шагах «Поиск шаблонов» и «Инстанцирование» производится полный перебор допустимых применений шаблонов из $\mathcal{P}$. Следовательно, траектория π_R будет сгенерирована как один из кандидатов.

На этапе «Верификация» траектория π_R будет проверена на валидность:

$$(\text{IsValid}(\text{Apply}(s_0, \pi_R)) = true)$$

и на прогресс по метрике μ.

Поскольку π_R ведет к цели (где $\mu = 0$), условие прогресса выполняется. Следовательно, оператор R выберет и применит эту траекторию или другую, также ведущую в $S_{intended}$. В виду того, что целевое состояние $s^* \in S_{intended}$ характеризуется отсутствием нарушений ($\mu(s^*) = 0$), условие строгого улучшения метрики $\mu(s^*) < \mu(s_0)$ выполняется автоматически, так как $\mu(s_0) > 0$).

Вывод: задача восстановления разрешима.

В интересах доказательства необходимости условия 1 (*X*-адаптивность) предположим обратное: оператор R работает успешно, но система не является *X*-адаптивной. Это означает, что существует состояние $s_{bad} \in S_{incident}$, из которого в графе $\mathcal{G}$ нет физического пути в $S_{intended}$, то есть s_{bad} – тупиковое состояние или изолированная компонента.

Оператор R генерирует только структурно корректные изменения, соответствующие дугам графа $\mathcal{G}$. Если пути в графе не существует, R не может его синтезировать траекторию восстановления, не нарушив валидность модели (предикат IsValid). Следовательно, $R(s_{bad})$ не достигнет цели, что противоречит условию успешности R.

Противоречие: значит, система обязана быть *X*-адаптивной.

В интересах доказательства необходимости условия 2 (Полнота $\mathcal{P}$) предположим обратное: оператор R работает успешно, система *X*-адаптивна (путь существует), но база $\mathcal{P}$ не полна. Это означает, что существует состояние $s \in S_{incident}$ и путь восстановления π, но в базе $\mathcal{P}$ нет шаблона, способного породить этот путь π или любой альтернативный путь к цели.

Оператор R работает только путем инстанцирования шаблонов из $\mathcal{P}$. Если нужного шаблона нет, множество кандидатов на шаге «Инстанцирование» не будет содержать решения, ведущего в $S_{intended}$. Оператор вернет исходное состояние s или выполнит частичное улучшение и застрянет в локальном минимуме, не достигнув цели. Это противоречит условию, что R решает задачу.

Противоречие: значит, база $\mathcal{P}$ обязана быть полной.

Заключение. Теорема доказана полностью. Условия структурной связности и информационной полноты являются необходимыми и достаточными для алгоритмической разрешимости задачи.

Отметим следующие особенности подхода:

– инвариантность к источнику (управляющее воздействие формируется на основе анализа графа $\mathcal{G}$, а не на основе идентификации причины сбоя);

– многовариантность (возврат осуществляется в любое достижимое состояние из $S_{intended}$);

– гарантия адаптации (возможность восстановления обеспечивается наличием структурно разре-

шенных путей в графе, существование которых постулируется свойством *X*-адаптивности);

– кросс-уровневая восстанавливаемость (связность графа $\mathcal{G}$ позволяет решать проблему на одном уровне средствами других (функциональная избыточность)).

Таким образом, *X*-адаптивность является строгим системным свойством, гарантирующим принципиальную возможность нейтрализации дестабилизирующих факторов произвольной природы.

4.4. Свойства оператора *R*

На основе доказанных утверждений и алгоритмической структуры оператор *R* обладает определенными формальными свойствами.

1) Безопасность (*Safety*): оператор сохраняет инвариант валидности (Утверждения 1).

2) Относительная полнота ($\mathcal{P}$-*Completeness*): если существует целевое состояние $s^* \in S_{intended}$, достижимое через композицию шаблонов из $\mathcal{P}$, то оператор найдет соответствующую траекторию (Утверждения 2).

3) Завершаемость (*Termination*): каждый вызов *R* завершается за конечное число шагов.

4) Строгая монотонность сходимости (*Strict Monotonicity*): для любого успешного управляющего воздействия выполняется условие строгого уменьшения метрики отклонения, математически гарантирует отсутствие зацикливания (*livelock*) в процессе адаптации:

$$R(s) \neq s \Longrightarrow \mu\big(R(s)\big) < \mu(s).$$

5) Идемпотентность в целевом множестве: если система уже находится в штатном состоянии, оператор не вносит изменений.

6) Иерархическая согласованность: все генерируемые изменения топологически корректны относительно HOP, учитывая направленность влияния между уровнями.

Это соответствует парадигме «fail-safe»: в худшем случае, если решение не найдено в базе $\mathcal{P}$, то система остается в исходном инцидентном состоянии $R(s) = s$ и инициирует эскалацию, но никогда не переходит в невалидное или хаотическое состояние.

Все изменения строго регламентированы: вертикальные воздействия (назначение прав) идут сверху вниз, горизонтальные – внутри уровня.

4.5. Аналитические следствия: иерархия как фактор уязвимости и защиты

Формальная структура модели ONYX позволяет строго доказать два фундаментальных свойства, характеризующих природу современных гибридных угроз.

Следствие 1. Эффект «Конуса влияния» (*Cone of Influence*)

В силу транзитивности управляющих связей в иерархии ($Ens \to P \to Hard \to Soft$) мощность атаки $Power(v)$, определяемая как множество вершин, достижимых из захваченного узла v, монотонно возрастает с повышением уровня иерархии.

Атакующий, скомпрометировавший узел на уровне *Soft*, например, *SQL*-инъекция, ограничен рамками программной среды. Напротив, атакующий на уровне *Ens*, например, изменение бюджетной политики, инициирует каскадные изменения на всех нижележащих уровнях. Таким образом, уровень иерархии выступает мультипликатором ущерба: захват управления (*Ens*, *P*) всегда опаснее захвата исполнения (*Hard*, *Soft*).

Следствие 2. Принцип уровневой нейтрализации

Модель разграничивает понятия компенсации и нейтрализации:

– компенсация (обеспечение живучести) возможна средствами нижестоящих уровней (например, при отказе оборудования (*Hard*) программный уровень (*Soft*) может снизить нагрузку, сохранив валидность системы ($s \in S_{valid}$), но не устранив причину сбоя);

– нейтрализация (восстановление исходной структуры) возможна только средствами того же или более высокого уровня (в силу иерархичности нижестоящий уровень не имеет управляющих связей к вышестоящему; следовательно, дефект уровня ℓ_i не может быть исправлен командой с уровня ℓ_{i+k}).

Анализ кейса. Социальная инженерия как атака «Вертикальной инъекции»

В контексте ONYX социальная инженерия представляет собой атаку, эксплуатирующую свойство иерархического доверия. Злоумышленник атакует уровень *P* (персонал), получая контроль над узлом v_{admin}, обладающим легитимными правами управления подграфом $G_{sub} \subset (Hard \cup Soft)$. Технические средства защиты уровней *Hard* и *Soft* конструктивно не способны нейтрализовать такую атаку, так как согласно предикату IsValid команды от v_{admin} являются валидными.

Из модели ONYX следует, что парирование такой угрозы возможно исключительно горизонтально (на уровне *P*) или вертикально (с уровня *Ens*): соответственно, введением топологических ограничений вида «правило двух ключей» (подтверждение действия двумя независимыми узлами *P*) или изменением мета-политик, блокирующих определенные типы транзакций для всех сотрудников.

5. Ограничения модели

Несмотря на доказанные свойства алгоритмической разрешимости и инвариантности, предложенная модель ONYX имеет ряд ограничений.

Во-первых, эпистемологическое ограничение базы шаблонов (проблема атак «нулевого дня»). Согласно Теореме 2, полнота восстановления прямо зависит от репрезентативности конечного множества $\mathcal{P}$. Если система сталкивается с принципиально новым вектором гибридной атаки или каскадным сбоем, для парирования которого в базе знаний нет подходящего структурного паттерна (предусловия π_k), оператор R не сможет синтезировать валидную траекторию. В этом случае модель переходит в режим эскалации инцидента оператору-человеку, гарантируя ненанесение ущерба, но теряя свойство автономности.

Во-вторых, вычислительная сложность в реальных масштабах (Enterprise-scale). Хотя проверка предиката IsValid(s) относится к классу LOGSPACE по сложности данных, на практике информационные системы корпоративного уровня представляют собой графы с десятками миллионов вершин и ребер. В условиях лавинного потока инцидентов, например, при массированной DDoS-атаке, полный перебор допустимых применений шаблонов на этапе инстанцирования может привести к вычислительным задержкам, превышающим допустимое время реакции. Это потребует применения эвристик для сужения пространства поиска ΔS, перехода к графовым нейронным сетям для аппроксимации оператора R или перехода от дискретной природы модели к непрерывной, чтобы иметь возможность применять методы оптимизации, например, градиентный спуск.

В-третьих, уязвимость мета-уровня. Модель предполагает, что онтология $\mathcal{L}$ и сами правила валидации (TGD и EGD) неизменны и защищены. Если злоумышленник, например, высокопривилегированный инсайдер, компрометирует сам инструмент адаптивного управления, изменяя предикаты или семантику базы $\mathcal{P}$, гарантии безопасности (Утверждение 1) аннулируются, так как нарушается аксиоматический базис модели.

6. Квалиметрическая оценка модели ONYX

ONYX – созидательная архитектурная модель: она проектирует новую структуру управления ИС, а не описывает существующую. Ее качество оценивается не по адекватности, а по пригодности – способности решать задачу адаптивного управления в условиях ДФ [23].

Показатели качества ONYX:

1) пригодность; ONYX, в отличие от фрагментарных подходов, интегрирует произвольное число уровней (в статье на примере обеспечивающего, персонала, аппаратного и программного уровней) в единый формализованный граф, обеспечивая упреждающее обнаружение и коррекцию структурных и функциональных дисбалансов;

2) развиваемость; иерархия ONYX масштабируема – допускает введение новых уровней, например, виртуальной инфраструктуры, или агрегацию существующих;

3) соответствие стандартам; модель формализует требования стандартов[9] через единый механизм состояний и переходов;

4) вычислимость; предикаты IsValid(s) и IsIntended(s) проверяются за полиномиальное время, оператор R гарантирует сохранение валидности – что обеспечивает практическую применимость.

Элементы модели ONYX апробированы при создании киберполигона МЧС России [24], проектировании фреймворка обоснования процедур мониторинга и реагирования на инциденты ИБ [25].

Пример. Рассмотрим инцидент на аппаратном уровне (*Hard* (ввод резерва) – на рисунке 1 слева): в результате DDoS-атаки эффективная пропускная способность основного канала связи v_{link} падает со 100 до 5 Мбит/с. При этом критический сервис v_{srv} (уровень *Soft* (сжатие данных), снижение требования к каналу – на рисунке 1 справа) требует поток данных $Demand = 20$ Мбит/с.

Состояние переходит в $s_1 \in S_{incident}$, так как нарушено условие валидности (параметрическое ограничение):

$$\text{prop}(v_{link},\text{'bandwidth'}) \geq$$
$$\geq \text{prop}(v_{srv},\text{'required\_bandwidth'}).$$

Оператор R, анализируя граф, генерирует две альтернативные траектории восстановления:

1) внутриуровневую (средствами *Hard*); активация «спящего» резервного канала связи v_{backup} с пропускной способностью 100 Мбит/с и переключение маршрутизации; атрибут пропускной способности восстанавливается;

– кросс-уровневую (средствами *Soft*); в условиях физической невозможности подключения резерва; оператор R инстанцирует на программном уровне модуль сжатия трафика $v_{compressor}$; это действие модифицирует атрибуты сервиса, снижая его потребность в канале до $Demand' = 4$ Мбит/с.

Таким образом, неравенство восстанавливается ($5 \geq 4$), и система возвращается в штатный режим функционирования за счет программной адаптации к аппаратной деградации. Следовательно, ONYX является первой моделью, обеспечивающей упреждающее, X-адаптивное управление произвольным количеством уровней ИС как единым целым.

[9] ISO/IEC 27001:2022 Clause 5, 7.1, ГОСТ Р ИСО/МЭК 27003-2012 Clause 7.1, ГОСТ Р ИСО/МЭК 27021-2021, ISO/IEC 27001:2022 Clause 7.2, ISO/IEC 27001:2022 Annex A.8 Asset Management, COBIT, ISO/IEC 27001 A.7–A.8, COBIT Technology

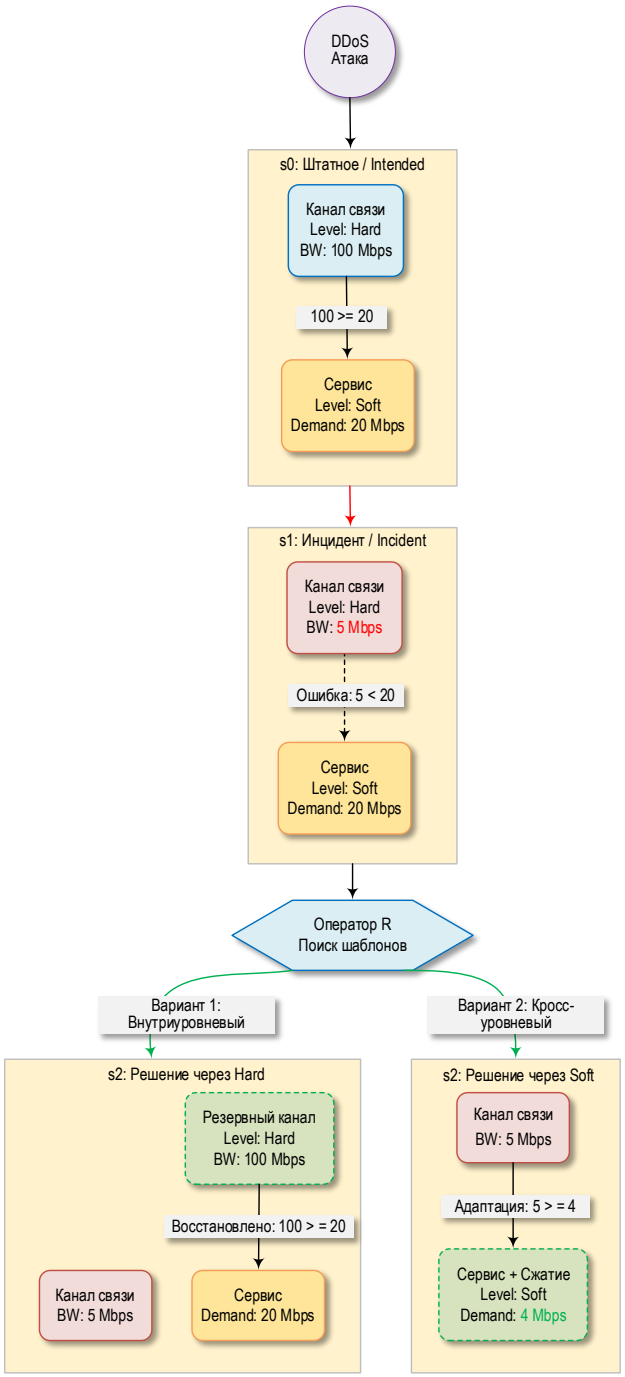

Рис. 1. Пример парирования параметрического возмущения: падение пропускной способности

Fig. 1. Illustration of Cross-Level Mitigation for a Bandwidth Reduction Incident

7. Обсуждение

База шаблонов $\mathcal{P}$ в текущей реализации задается экстенсионально, но формализм допускает ее интенсиональное расширение, через мета-шаблоны или правила, сгенерированные методами машинного обучения. Это открывает путь к автоматическому синтезу стратегий адаптации и станет предметом дальнейших исследований.

Практическая реализация требует: создания унифицированной шины данных для построения графа из гетерогенных источников (CMDB, IdM, Asset Management); калибровки человеческого фактора по реальным метрикам; разработки гибридного оператора R, сочетающего автоматизацию и поддержку принятия решений.

Ключевое преимущество ONYX перед традиционными SOAR-подходами – математическая гарантия безопасности: каждое управляющее воздействие верифицируется на соответствие IsValid(s), что исключает «дружественный огонь» при автоматическом реагировании.

Гибкость онтологии $\mathcal{L}$ позволяет применять архитектуру ONYX не только к агрегированным классам ресурсов аппаратного или программного обеспечения, и к детализированным техническим стекам, таким как сетевая модель OSI. В этом случае стандартные 7 уровней OSI дополняются уровнями «Пользователь» и «Политика», что позволяет создать систему управления, охватывающую полный жизненный цикл передачи информации – от физической среды до бизнес-решения.

8. Заключение

В статье представлена архитектурная модель X-адаптивного управления информационной системой ONYX. В отличие от существующих подходов к управлению информационными системами, которые, как правило, либо ограничиваются отдельными уровнями (программно-аппаратным, нормативно-правовым, персонала и т. д.), либо опираются на предзаданные модели угроз и эвристические процедуры реагирования, впервые предложен подход, научная новизна которого заключается в следующем:

1) универсальность формализма (разработанный математический аппарат (графовая динамика с ограничениями FOL) инвариантен к конкретной семантике уровней; введенный автором HOP определяет класс рассматриваемых систем, однако сам формализм ONYX применим для управления неиерархическими (mesh, роевыми) и гибридными системами при замене HOP на соответствующие топологические ограничения);

2) теоретико-множественные условия адаптации (сформулированы и доказаны два необходимых и достаточных условия разрешимости задачи восстановления: структурное (принадлежность инцидента области притяжения целевого множества) и информационное (полнота базы шаблонов); это переводит задачу адаптации из области эвристик в область строгой алгоритмической разрешимости);

3) аналитические следствия иерархии (на основе топологии графа доказан эффект «Конуса влияния», демонстрирующий, что уровень иерархии

является мультипликатором ущерба; формально обоснована невозможность нейтрализации атак на верхние уровни, например, социальная инженерия против персонала средствами нижних уровней (ПО и оборудование), что требует реализации механизмов «горизонтального контроля» или мета-управления);

4) кросс-уровневая восстанавливаемость (показано, что связность графа состояний позволяет реализовывать функциональную избыточность: дефицит ресурсов на одном уровне, например, аппаратный сбой, может быть компенсирован структурной перестройкой на другом (программная адаптация), что расширяет границы живучести системы);

5) гарантированная безопасность управления (введены понятия инварианта валидности и относительной полноты оператора R, что обеспечивает математическую гарантию того, что автоматическая реакция системы никогда не приведет к нарушению политик безопасности или целостности архитектурного замысла).

Практическая значимость модели состоит в создании фундамента для автоматизированных систем управления безопасностью, способных работать в условиях высокой неопределенности. Модель позволяет обнаруживать отклонения от целевого замысла на ранних стадиях (упреждающее управление) и генерировать верифицированные сценарии реагирования, объединяя организационные, кадровые и технические меры в единый контур управления. Перспективы развития связаны с внедрением методов машинного обучения для автоматического синтеза базы шаблонов $\mathcal{P}$, что позволит обеспечить полную автономность адаптации.

Список источников

1. Максимова Е.А. Аксиоматика инфраструктурного деструктивизма субъекта критической информационной инфраструктуры // Информатизация и связь. 2022. № 1. С. 68–74. DOI:10.34219/2078-8320-2022-13-1-68-74. EDN:ZMOPQB

2. Гурина Л.А., Томин Н.В. Интеллектуальные методы обеспечения кибербезопасности мультиагентных систем управления микросетями // Вопросы кибербезопасности. 2024. № 6(64). С. 53–64. DOI:10.21681/2311-3456-2024-6-53-64. EDN:BORTZT

3. Wei L., Yang Y., Wu J., Long C., Li B. Trust Management for Internet of Things: A Comprehensive Study // IEEE Internet of Things Journal. 2022. Vol. 9. Iss. 10. PP. 7664–7679. DOI:10.1109/JIOT.2021.3139989. EDN:JRPLEV

4. Nguyen T.T., Reddi V.J. Deep Reinforcement Learning for Cyber Security // IEEE Transactions on Neural Networks and Learning Systems. 2021. Vol. 34. Iss. 8. PP. 3779–3795. DOI:10.1109/TNNLS.2021.3121870. EDN:FKCUDV

5. Yin Z., Lin Y., Zhang Y., Qian Y., Shu F., Li J. Collaborative Multiagent Reinforcement Learning Aided Resource Allocation for UAV Anti-Jamming Communication // IEEE Internet of Things Journal. 2022. Vol. 9. Iss. 23. PP. 23995–24008. DOI:10.1109/jiot.2022.3188833. EDN:CQKPMR

6. Wan Z., Cho J.H., Zhu M., Anwar A.H., Kamhoua C.A., Singh M.P. Resisting Multiple Advanced Persistent Threats via Hypergame-Theoretic Defensive Deception // IEEE Transactions on Network and Service Management. 2023. Vol. 20. Iss. 3. PP. 3816–3830. DOI:10.1109/tnsm.2023.3240366. EDN:ISRRNW

7. Segovia-Ferreira M., Rubio-Hernan J., Cavalli A., Garcia-Alfaro J. A survey on cyber-resilience approaches for cyber-physical systems // ACM Computing Surveys. 2024. Vol. 56. Iss. 8. PP. 1–37. DOI:10.1145/3652953

8. Kuikka V., Rantanen H. Resilience of Multi-Layer Communication Networks // Sensors. 2022. Vol. 23. Iss. 1. P. 86. DOI:10.3390/s23010086. EDN:JTEMYM

9. Wang X., Niu B., Shang Z., Niu Y. Distributed resilient adaptive consensus tracking control of nonlinear multi-agent systems dealing with deception attacks via K-filters approach // Automatica. 2024. Vol. 169. P. 111871. DOI:10.1016/j.automatica.2024.111871. EDN:SLWBOM

10. Canonico R., Sperlì G. Industrial cyber-physical systems protection: A methodological review // Computers & Security. 2023. Vol. 135. P. 103531. DOI:10.1016/j.cose.2023.103531. EDN:SZERNJ

11. Arauz T., Chanfreut P., Maestre J.M. Cyber-security in networked and distributed model predictive control // Annual Reviews in Control. 2022. Vol. 53. PP. 338–355. DOI:10.1016/j.arcontrol.2021.10.005. EDN:DOVLWJ

12. Xing W., Shen J. Security Control of Cyber–Physical Systems under Cyber Attacks: A Survey // Sensors. 2024. Vol. 24. Iss. 12. P. 3815. DOI:10.3390/s24123815. EDN:OXISLH

13. Luo M., Yu Z., Xiao Y., Xiong L., Xu Q., Ma L., Wu Z. Full-order adaptive sliding mode control with extended state observer for high-speed PMSM speed regulation // Scientific Reports. 2023. Vol. 13. Iss. 1. P. 6200. DOI:10.1038/s41598-023-33455-x. EDN:SQOMRD

14. Полтавцева М.А. Модель активного мониторинга как основа управления безопасностью промышленных киберфизических систем // Вопросы кибербезопасности. 2021. № 2(42). С. 51–60. DOI:10.21681/2311-3456-2021-2-51-60. EDN:WSMBXF

15. Blanco C., Rosado D.G., Varela-Vaca Á.J., Gómez-López M.T., Fernández-Medina E. Onto-CARMEN: Ontology-driven approach for Cyber-Physical System Security Requirements meta-modelling and reasoning // Internet of Things. 2023. Vol. 24. P. 100989. DOI:10.1016/j.iot.2023.100989. EDN:HYOTME

16. Bakirtzis G., Sherburne T., Adams S., Horowitz B.M., Beling P.A., Fleming C.H. An ontological metamodel for cyber-physical system safety, security, and resilience coengineering // Software and Systems Modeling. 2022. Vol. 21. Iss. 1. PP. 113–137. DOI:10.1007/s10270-021-00892-z. EDN:VGYIJO

17. Kayan H., Nunes M., Rana O., Burnap P., Perera C. Cybersecurity of Industrial Cyber-Physical Systems: A Review // ACM Computing Surveys (CSUR). 2022. Vol. 54. Iss. 11s. P. 229. DOI:10.1145/3510410

18. Rahman M.H., Shafae M. Cyber-Physical Security Vulnerabilities Identification and Classification in Smart Manufacturing: A Defense-in-Depth Driven Framework and Taxonomy // Journal of Computing and Information Science in Engineering. 2025. Vol. 25. Iss. 9. P. 091005. DOI:10.1115/1.4068844. EDN:OKVQFQ

19. Damm W., Hess D., Schweda M., Sztipanovits J., Bengler K., Biebl B. et al. A Reference Architecture of Human Cyber-Physical Systems–Part I: Fundamental Concepts // ACM Transactions on Cyber-Physical Systems. 2024. Vol. 8. Iss. 1. PP. 1–32. DOI:10.1145/3622879

20. Immerman N. Descriptive Complexity // Graduate Texts in Computer Science. New York, Berlin, Heidelberg: Springer-Verlag, 1999. DOI:10.1007/978-1-4612-0539-5

21. Грызунов В.В. Модель целенаправленных агрессивных действий на информационно-вычислительную систему // Труды Третьей международной научно-практической конференции «Человеческий фактор в сложных технических системах и средах» (ЭРГО-2018, Санкт-Петербург, Российский Федерация, 07.06.2018). Тверь: Межрегиональная общественная организация "Эргономическая ассоциация", 2018. С. 300–305. EDN:YUORVZ

22. Gryzunov V.V. Conceptual Model for Adaptive Control of a Geographic Information System under Conditions of Destabilization // Automatic Control and Computer Sciences. 2021. Vol. 55. Iss. 8. PP. 1222–1227. DOI:10.3103/S0146411621080381. EDN:KWLWGO

23. Микони С.В., Соколов Б.В., Юсупов Р.М. Квалиметрия моделей и полимодельных комплексов: монография. М.: РАН, 2018. 314 с. DOI:10.31857/S9785907036321000001. EDN:VVUKQW

24. Грызунов В.В., Шестаков А.В. Модель системы адаптивного управления киберполигоном МЧС России на основе операторного уравнения // Вопросы кибербезопасности. 2024. № 6(64). С. 140–149. DOI:10.21681/2311-3456-2024-6-140-149. EDN:GSHMNZ

25. Грызунов В.В., Шестаков А.В. Многоуровневый фреймворк обоснования процедур мониторинга и реагирования на инциденты информационной безопасности // Вопросы кибербезопасности. 2025. № 6(70). С. 14–24. DOI:10.21681/2311-3456-2025-6-14-24. EDN:HTFRHK

References

1. Maksimova E.A. Axiomatics of infrastructural destructivism of a critical information infrastructure subject. *Informatization and communication*. 2022;1:68–74. (in Russ.) DOI:10.34219/2078-8320-2022-13-1-68-74. EDN:ZMOPQB

2. Gurina L.A., Tomin N.V. Intelligent methods of ensuring cybersecurity of multi-agent microgrid control systems. *Voprosy kiberbezopasnosti (Cybersecurity Issues)*. 2024;6(64):53–64. (in Russ.) DOI:10.21681/2311-3456-2024-6-53-64. EDN:BORTZT

3. Wei L., Yang Y., Wu J., Long C., Li B. Trust Management for Internet of Things: A Comprehensive Study. *IEEE Internet of Things Journal*. 2022;9(10):7664–7679. DOI:10.1109/JIOT.2021.3139989. EDN:JRPLEV

4. Nguyen T.T., Reddi V.J. Deep Reinforcement Learning for Cyber Security. *IEEE Transactions on Neural Networks and Learning Systems*. 2021;34(8):3779–3795. DOI:10.1109/TNNLS.2021.3121870. EDN:FKCUDV

5. Yin Z., Lin Y., Zhang Y., Qian Y., Shu F., Li J. Collaborative Multiagent Reinforcement Learning Aided Resource Allocation for UAV Anti-Jamming Communication. *IEEE Internet of Things Journal*. 2022;9(23):23995–24008. DOI:10.1109/jiot.2022.3188833. EDN:CQKPMR

6. Wan Z., Cho J.H., Zhu M., Anwar A.H., Kamhoua C.A., Singh M.P. Resisting Multiple Advanced Persistent Threats via Hypergame-Theoretic Defensive Deception. *IEEE Transactions on Network and Service Management*. 2023;20(3):3816–3830. DOI:10.1109/tnsm.2023.3240366. EDN:ISRRNW

7. Segovia-Ferreira M., Rubio-Hernan J., Cavalli A., Garcia-Alfaro J. A survey on cyber-resilience approaches for cyber-physical systems. *ACM Computing Surveys*. 2024;56(8):1–37. DOI:10.1145/3652953

8. Kuikka V., Rantanen H. Resilience of multi-layer communication networks. *Sensors*. 2022;23(1):86. DOI:10.3390/s23010086. EDN:JTEMYM

9. Wang X., Niu B., Shang Z., Niu Y. Distributed resilient adaptive consensus tracking control of nonlinear multi-agent systems dealing with deception attacks via K-filters approach. *Automatica*. 2024;169:111871. DOI:10.1016/j.automatica.2024.111871. EDN:SLWBOM

10. Canonico R., Sperlì G. Industrial cyber-physical systems protection: A methodological review. *Computers & Security*. 2023;135:103531. DOI:10.1016/j.cose.2023.103531. EDN:SZERNJ

11. Arauz T., Chanfreut P., Maestre J.M. Cyber-security in networked and distributed model predictive control. *Annual Reviews in Control*. 2022;53:338–355. DOI:10.1016/j.arcontrol.2021.10.005. EDN:DOVLWJ

12. Xing W., Shen J. Security Control of Cyber–Physical Systems under Cyber Attacks: A Survey. *Sensors*. 2024;24(12):3815. DOI:10.3390/s24123815. EDN:OXISLH

13. Luo M., Yu Z., Xiao Y., Xiong L., Xu Q., Ma L., Wu Z. Full-order adaptive sliding mode control with extended state observer for high-speed PMSM speed regulation. *Scientific Reports*. 2023;13(1):6200. DOI:10.1038/s41598-023-33455-x. EDN:SQOMRD

14. Poltavtseva M. Active monitoring model as a basis for security management of industrial CPS. *Voprosy kiberbezopasnosti (Cybersecurity Issues)*. 2021;42(2):51–60. (in Russ.) DOI:10.21681/2311-3456-2021-2-51-60. EDN:WSMBXF

15. Blanco C., Rosado D.G., Varela-Vaca Á.J., Gómez-López M.T., Fernández-Medina E. Onto-CARMEN: Ontology-driven approach for Cyber–Physical System Security Requirements meta-modelling and reasoning. *Internet of Things*. 2023;24:100989. DOI:10.1016/j.iot.2023.100989. EDN:HYOTME

16. Bakirtzis G., Sherburne T., Adams S., Horowitz B.M., Beling P.A., Fleming C.H. An ontological metamodel for cyber-physical system safety, security, and resilience coengineering. *Software and Systems Modeling*. 2022;21(1):113–137. DOI:10.1007/s10270-021-00892-z. EDN:VGYIJO

17. Kayan H., Nunes M., Rana O., Burnap P., Perera C. Cybersecurity of industrial cyber-physical systems: A review. *ACM Computing Surveys (CSUR)*. 2022;54(11s):229. DOI:10.1145/3510410

18. Rahman M.H., Shafae M. Cyber-Physical Security Vulnerabilities Identification and Classification in Smart Manufacturing: A Defense-in-Depth Driven Framework and Taxonomy. *Journal of Computing and Information Science in Engineering*. 2025:25(9):091005. DOI:10.1115/1.4068844. EDN:OKVQFQ

19. Damm W., Hess D., Schweda M., Sztipanovits J., Bengler K., Biebl B. et al. A Reference Architecture of Human Cyber-Physical Systems–Part I: Fundamental Concepts. *ACM Transactions on Cyber-Physical Systems*. 2024;8(1):1–32. DOI:10.1145/3622879

20. Immerman N. *Descriptive Complexity*. Graduate Texts in Computer Science. New York, Berlin, Heidelberg: Springer-Verlag; 1999. DOI:10.1007/978-1-4612-0539-5

21. Gryzunov V.V. Model of Purpose Aggressive Actions on the Information-Computing System. *Proceedings of the Third International Scientific and Practical Conference on Human Factor in Complex Technical Systems and Environments, ERGO-2018, 07.06.2018, St. Petersburg, Russian Federation*. Tver: Ergonomic Association Publ.; 2018. p.300–305. (in Russ.) EDN:YUORVZ

22. Gryzunov V.V. Conceptual Model for Adaptive Control of a Geographic Information System under Conditions of Destabilization. *Automatic Control and Computer Sciences*. 2021;55(8):1222–1227. DOI:10.3103/S0146411621080381. EDN:KWLWGO

23. Mikoni S.V., Sokolov B.V., Yusupov R.M. *Qualimetry of Models and Polymodel Complexes*. Moscow: RAS Publ.; 2018. 314 p. (in Russ.) DOI:10.31857/S9785907036321000001. EDN:VVUKQW

24. Gryzunov V.V., Shestakov A.V. Model of the adaptive control system of the cyber range of the russian emergencies ministry based on the operator equation. *Voprosy kiberbezopasnosti (Cybersecurity Issues)*. 2024;64(6):140–149. (in Russ.) DOI:10.21681/2311-3456-2024-6-140-149. EDN:GSHMNZ

25. Gryzunov V.V., Shestakov A.V. A multi-level framework for justifying information security incident monitoringand response procedures. *Voprosy kiberbezopasnosti (Cybersecurity Issues)*. 2025;6(70):14–24. (in Russ.) DOI:10.21681/2311-3456-2025-6-14-24. EDN:HTFRHK

Статья поступила в редакцию 10.01.2026; одобрена после рецензирования 18.03.2026; принята к публикации 17.04.2026
The article was submitted 10.01.2026; approved after reviewing 18.03.2026; accepted for publication 17.04.2026

Информация об авторе:

ГРЫЗУНОВ Виталий Владимирович — доктор технических наук, доцент, профессор кафедры прикладной информатики и безопасности информационных технологий Санкт-Петербургского университета ГПС МЧС России
https://orcid.org/0000-0003-4866-217X

Автор сообщает об отсутствии конфликтов интересов.
The author declares no conflicts of interests.