Specifying Deception Requirements with iStar

Ricardo Antônio Rebouças Celestino[1], Jhonatan de Sousa Carvalho[1], Enyo Gonçalves[2], Paulo Henrique Mendes Maia[1]

[1]State University of Ceará, Fortaleza, Ceará, Brazil

[2]Federal University of Ceará, Quixadá, Ceará, Brazil

ricardo.celestino@gmail.com,
jhonatan.carvalho@aluno.uece.br, enyo@ufc.br,
pauloh.maia@uece.br

Abstract. Cybersecurity plays a central role in software development. Deception-based defense is a cybersecurity subfield in which actions are intentionally employed to cause misrepresentation and induce erroneous inferences on attackers. Deception can be employed at different levels of computation, from network to application-level, which demands careful planning and coordination between multiple strategies and tactics. The specification of such systems is essential to the successful development of software projects. This paper proposes a goal-oriented requirements engineering approach for representing systems with cyber deception requirements, called iStar4Deception. The development of this approach followed the PRISE process to ensure conceptual rigor, compatibility with native iStar, and the reuse of well-established extensions documented in the iStar Extension Catalog and supported by piStar-EXT. To demonstrate the applicability of iStar4Deception, we conducted a real-world case study of a Brazilian bank slip payment system in which cyber deception mechanisms and automated redirection of malicious traffic are strategically deployed not only to detect, delay, and analyze attacks but also to preserve critical assets. The results show that the proposed extension enables a systematic modeling of deception-oriented goals, dependencies, and qualities, providing analysts with a structured approach to integrating cyber deception strategies into goal-oriented requirements models.

1 Introduction

Information security [ISO/IEC, 2022] plays a central role in enforcing the three pillars established by NIST (Confidentiality, Integrity, and Availability) ensuring that data and systems are protected against threats, remain complete, and are accessible only to authorized users [NIST, 1977]. These pillars are supported by several mechanisms, such as firewalls, intrusion detection systems, antivirus software, security policies, backups, employee training, and deception techniques. Real-world cases demonstrate the impact that vulnerabilities in computer systems can have when security is not adequately addressed during the development cycle. Over the last decades, cyberattacks have affected organizations such as the U.S. Department of Defense, NASA, Yahoo, the PlayStation Network, and Saudi Aramco, resulting in financial losses and operational instability [Cobalt Blog, 2024].

Deception techniques represent one of the few defensive approaches capable of exploiting attacker vulnerabilities, as they aim to manipulate human perception by exploiting psychological vulnerabilities — which directly impact beliefs, decisions, and actions [Mitnick and Simon, 2002]. These techniques consist of creating false or misleading scenarios in order to manipulate the attacker. A common example is the use of honeypots, which simulate vulnerable systems to attract and monitor malicious activity. These mechanisms are widely used in cybersecurity as an effective defensive strategy. Their effectiveness lies in the ability to capture detailed information about attacker behavior, attack vectors, and exploitation techniques, often producing few false positives. Studies using distributed honeypot infrastructures have shown it is possible to systematically analyze real-world attack patterns, including large-scale automated campaigns targeting specific services [Ceron et al., 2013].

For software development, certain disciplines must be carried out, such as Requirements Engineering (RE), modeling, design and implementation, testing, and evolution [Sommerville, 2011]. In the Requirements Engineering phase [Kotonya and Sommerville, 1998], where the objective of the software is defined, if the defined requirements are inadequate, incomplete, ambiguous, or inconsistent, there will be a significant impact on the quality of the software [Lapouchnian, 2005].

In Requirements Engineering (RE), several methodologies can be adopted, such as goal-oriented, scenario-oriented, and object-oriented approaches. Among them, Goal-Oriented Requirements Engineering (GORE) has stood out in recent years for its ability to structure the requirements definition process based on organizational goals [van Lamsweerde, 2001]. The GORE approach uses goals to support elicitation, analysis, specification, negotiation, documentation, and requirements evolution. KAOS [van Lamsweerde, 2001] and iStar [Dalpiaz et al., 2016] are examples of GORE modeling languages. However, connecting Requirements Engineering and cyber deception requires modeling capabilities that go beyond those offered by traditional GORE approaches. In addition to capturing the goals of legitimate stakeholders, cyber deception demands explicit representation of adversarial intent, attackers' beliefs, and the deceptive mechanisms used to influence attacker behavior.

Given this context, the objective of this work is to support the requirements specification of deception-oriented security systems through an iStar extension, combining existing constructs with new ones that address the specific needs of this area.

This paper is organized as follows. Section 2 presents the theoretical background, introducing the iStar language, its main constructs and extensions, as well as fundamental concepts of cyber deception. Section 3 reviews related work, discussing existing approaches and extensions for modeling security and deception requirements. Section 4 describes the methodology adopted in this work. Section 5 presents the proposed iStar4Deception extension, detailing its development process, constructs, and design rationale. Section 6 demonstrates the application of the extension through a real-world case study of a Brazilian bank slip payment system. Finally, Section 7 concludes the paper and outlines directions for future work.

2 Background

2.1 iStar and its extensions

The iStar language, proposed by Yu [Yu, 1995], is a framework oriented toward goal-oriented and actor-oriented modeling, and is widely used in Goal-Oriented Requirements Engineering (GORE) [van Lamsweerde, 2001]. The language has evolved over time, and in 2016, the iStar 2.0 version was introduced [Dalpiaz et al., 2016], providing greater clarity and consistency in both notation and semantics. This version defines two main models: the Strategic Dependency (SD) model and the Strategic Rationale (SR) model, along with formal rules to ensure consistency and avoid ambiguity.

iStar is structured around three categories of constructs: actors, intentional elements, and links that represent relationships between them. Actors may represent people, organizations, or systems that depend on one another to achieve goals, perform tasks, or obtain resources. Actors are divided into two subtypes: **agent**, which represents a concrete entity such as a person or system, and **role**, which represents a function or responsibility that an agent may assume within an organizational context. Relationships between actors are primarily captured in the SD model and include generalization and specialization relations (Is-a), participation relations that indicate organizational membership or role playing (Participates-In), and dependency relations, in which one actor relies on another to achieve a goal, perform a task, provide a resource, or satisfy a quality, as illustrated in Figure 1.

Figure 1. iStar Constructs

The intentional elements in iStar encompass goals, which represent desired states an actor aims to achieve; qualities, which capture qualitative objectives without clearly defined satisfaction criteria; tasks, which correspond to actions executed to achieve goals; and resources, which represent physical or informational entities required for task execution.

Yu [Yu, 1995] defines four types of *dependum* that characterize dependencies between actors. In a **goal dependency**, the depender relies entirely on the dependee to achieve a specific goal, becoming vulnerable if the dependee fails. A **task dependency** occurs when the depender depends on the dependee to execute a task necessary for goal achievement, with failure similarly leading to vulnerability. In a **resource dependency**, the depender relies on the dependee to provide access to required resources, such as data or equipment, and becomes vulnerable if these resources are unavailable. Finally, a **quality (softgoal) dependency** involves reliance on the dependee to satisfy a qualitative objective, in which case the depender retains the discretion to accept or reject the achieved outcome.

According to Horkoff and Yu [Horkoff and Yu, 2016] and Dalpiaz et al. [Dalpiaz et al., 2016], iStar models can be expressed through two main perspectives. The

Strategic Dependency (SD) model provides a high-level view of the system, focusing on actors and the dependencies between them, and is typically used in early stages of requirements analysis. In contrast, the Strategic Rationale (SR) model offers an internal view of actors, detailing their goals, tasks, qualities, resources, and the reasoning processes that justify their decisions. The SR model commonly includes decomposition links, contribution links, and means–ends links to explain how goals are operationalized.

As the use of iStar expanded to more complex domains, researchers proposed several extensions to the language in order to address concerns not fully covered by the core notation. These extensions preserve the intentional modeling principles of iStar while introducing new constructs tailored to specific application contexts. Creating extensions in iStar is a common process frequently used by various authors across a wide range of fields, such as cybersecurity [Mouratidis et al., 2007], data warehousing [Morandini et al., 2017], and adaptive systems [Giorgini et al., 2005].

The cybersecurity field is one of the areas with the largest number of iStar extensions. According to Gonçalves et al. [Gonçalves et al., 2018], a total of 96 papers on iStar extensions were identified, of which 18 present extensions related to security in the context of iStar. Moreover, it is increasingly necessary to enhance security solutions to mitigate risks of attacks and intrusions, information theft, data hijacking and tampering, information leakage, among many other potential problems. Given the large number of new extensions, another problem becomes apparent: their integration with other projects in the iStar community; since most extensions do not take into account the need for integration with other projects. The lack of such integration can hinder the dissemination and use of these extensions [Franch et al., 2018]. PRISE (Process to Support iStar Extensions) provides methodological support for the systematic development of iStar extensions, ensuring consistency, interoperability, and reuse of existing constructs [Gonçalves, 2019]. Given this, PRISE proved essential in the development of our extension, ensuring that we did not unnecessarily duplicate existing extensions. In our case study, it was necessary to use extensions previously specified by [Elahi et al., 2010], where the author proposes a methodological framework for the elicitation and analysis of security requirements, focusing on vulnerabilities and offering modeling and analysis tools to help system designers understand vulnerabilities and their consequences.

In this work, we propose the extension iStar4Deception, which focuses on modeling deception strategies in security-sensitive systems. This extension introduces additional constructs to explicitly represent deceptive goals, misleading tasks, and decoy resources, allowing designers to model defensive mechanisms that intentionally confuse or mislead attackers. By extending the original iStar framework, iStar4Deception enables the systematic analysis of security strategies while maintaining compatibility with the SD and SR modeling perspectives.

Figure 2 illustrates an example of an SR model, adapted from [Elahi et al., 2010]. The model shows attacks that exploit vulnerabilities in the browser and the web server. In this model, several new constructs were created and represent malicious intentional elements. It also shows actions that can prevent or minimize the consequences of those attacks.

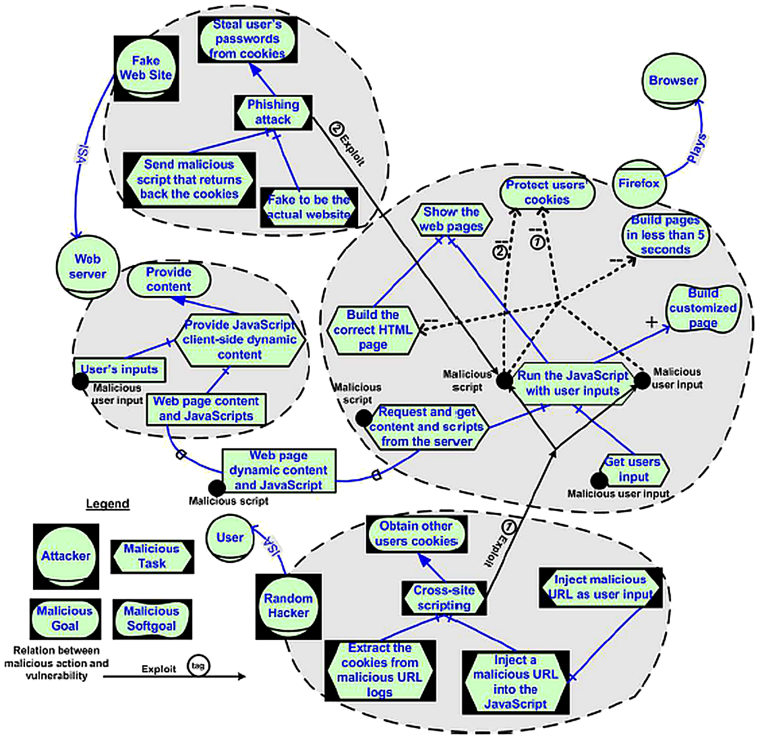

Figure 2. Attacker template view for the browser and web server example. [Elahi et al., 2010].

Modern cybersecurity increasingly requires adaptive security that can adjust controls and policies at runtime based on contextual information, thereby proactively mitigating risks [Pirmez et al., 2008]. Although focused on runtime adaptation, these approaches highlight the need to model context-aware security requirements early. This motivates our use of a goal-oriented approach (iStar) to capture dynamic dependencies, alternative strategies, and evolving security objectives, including cyber-deception mechanisms.

2.2 Deception

According to Whaley and Bell [Whaley and Bell, 1991], deception refers to deliberate actions designed to induce others into making incorrect inferences. In nature, deception appears through mimicry, where organisms confuse predators by altering their appearance, as discussed by Smith [Smith, 2005].

In the digital domain, these strategies are referred to as *cyber deception*. Climek et al. [Climek et al., 2016] define cyber deception as the deliberate use of uncertainty and misinformation to obscure networks, distort adversarial situational awareness, and mislead decision-making processes. Such techniques help defenders collect intelligence, introduce delays, increase the attacker's effort, and reveal adversarial tactics.

Cyber deception mechanisms have evolved significantly since the early 1990s, when research primarily focused on intrusion detection and early warning systems. Modern deception frameworks integrate high-interaction honeypots, honeytokens, decoys, and falsified system information designed to manipulate attacker perception [Spitzner, 2003].

Beyond these mechanisms, the literature highlights several key *conceptual foundations* that are essential for understanding deception in security systems and, consequently, for designing a modeling extension such as iStar4Deception:

Table 1. Conceptual Foundations for Deception

Concept	Definition
Deception Goals	Deception strategies are not arbitrary; they serve explicit objectives such as delaying attackers, diverting them from critical assets, exhausting their resources, or revealing their tactics and tools. These goals must be modeled clearly when designing a deception-based system [de Faveri, 2021].
Deception Assets	These include fake resources, falsified information, decoys, and simulated services that the attacker perceives as legitimate. Recent studies emphasize that deception assets must incorporate simulated user behavior and dynamic updates to maintain realism against skilled attackers [Pacheco and Staino, 2025]. Modeling deception assets is crucial because they serve as the operational means through which deception goals are achieved [Pacheco and Staino, 2025].
Attacker Profiling and Behavioral Modeling	Cyber deception relies on modeling attacker goals, capabilities, and beliefs to infer intent and deploy effective deceptive strategies [Kotenko et al., 2023]. In goal-oriented requirements engineering, explicitly representing these intentional perspectives helps identify interventions that effectively mislead adversarial decision-making [Shinde et al., 2021, Kotenko et al., 2023].
Misdirection Strategies	Common strategies include obscuring vulnerabilities and creating attractive false targets [Steingartner et al., 2021]. Modern deception frameworks extend beyond simple misdirection by incorporating adaptive analysis; by continuously monitoring attacker interactions with decoys, defenders can analyze adversarial behavior in real time, collect threat intelligence, and dynamically refine defensive strategies [Samhruth et al., 2025].
Engagement and Interaction Phases	Deception activities often unfold across phases such as attraction, engagement, misinformation, and evidence collection. Each phase has different requirements and success criteria that must be explicitly modeled [Pacheco and Staino, 2025].
Risk and Consequence Evaluation	Unlike traditional security controls, cyber deception intentionally exposes system components to attackers, introducing inherent risks. Formal and probabilistic models, such as Markov Decision Processes (MDPs), enable the evaluation of deception effectiveness, costs, and risks before deployment. In goal-oriented requirements engineering, these assessments help ensure that deception strategies do not compromise safety or operational availability. [Haseeb et al., 2022].

3 Related Work

This section reviews relevant related work on modeling security and deception requirements, focusing on goal-oriented and cybersecurity-oriented approaches. One of them is a KAOS extension of deception modeling and the other three are iStar extensions in cybersecurity.

De Faveri proposed an approach for modeling deception strategies in cybersecurity, offering a conceptual foundation for psychological and computational aspects of deception [de Faveri, 2021]. His proposal suggests an extension of the KAOS language to represent techniques, tactics, and goals related to deception. Similar to our work, De Faveri recognizes deception as a fundamental component in modern security. However, his approach is based on KAOS and does not follow a structured process such as PRISE. Even so, his conceptual contributions helped inspire our definition of deception-related concepts. To adapt these ideas to our context, we mapped KAOS constructs into the iStar paradigm.

Mouratidis [Mouratidis et al., 2007] proposed an extension of the Tropos method to model threats and security requirements in cloud-based systems. This line of work later evolved into *Secure Tropos*, introducing constructs such as *Security Goal*, *Security Mechanism*, *Threat*, and *Security Constraint*. While our work also addresses cybersecurity, its focus differs significantly: Secure Tropos aims at protecting systems from known threats, while iStar4Deception focuses on deception as a defensive strategy. Additionally, unlike Secure Tropos, our extension follows the PRISE guidelines, ensuring methodological rigor and standardization.

Elahi also proposed an iStar extension for security, focusing on modeling vulnerabilities and malicious agents [Elahi et al., 2010]. Their model introduces constructs such as *Malicious Goal*, *Malicious Task*, and *Attacker*, explicitly representing malicious intent and potential attack targets within the system. Part of these constructs inspired our extension: while Elahi's proposal focuses on identifying and mitigating risks, iStar4Deception builds upon similar concepts to propose mechanisms of manipulation and misinformation as defense strategies. Unlike our proposal, which explicitly follows the PRISE process, Elahi et al. do not report the use of a dedicated extension engineering methodology.

The iStar4Safety [Ribeiro et al., 2019] is an iStar extension for modeling safety-critical systems in domains such as transportation, energy, and healthcare. This extension introduces constructs like *Safety Goal* and *Hazard* and follows the PRISE process rigorously. iStar4Safety shares methodological foundations with our work but differs in thematic focus: while iStar4Safety targets operational safety and accident prevention, iStar4Deception addresses the active manipulation of attackers using deceptive information.

From a requirements engineering perspective, a key challenge in cybersecurity is aligning high-level security objectives with operational defense mechanisms. Traditional security modeling approaches, including threat modeling methods such as STRIDE and security-oriented frameworks such as Secure Tropos, primarily focus on identifying vulnerabilities, mitigating attack vectors, and enforcing security controls [Mouratidis et al., 2007; Elahi et al., 2010]. In contrast, cyber deception adopts a proactive defense strategy centered on manipulating attacker perceptions, influencing adversarial decision-making, and collecting threat intelligence. Existing approaches that support these objectives typically

rely on low-level techniques, such as Markov Decision Processes and runtime adaptation mechanisms [Haseeb et al., 2022; Pirmez et al., 2008], providing limited support for modeling deception during the early requirements engineering stages. iStar4Deception addresses this gap by introducing an intentional modeling layer that enables analysts to capture adversarial goals, deceptive strategies, and their relationships with security objectives, thereby supporting the systematic design of active cyber defense mechanisms.

4 Methodology

To select the most suitable framework for developing our extension, we considered the comparison by Werneck et al. [Werneck et al., 2009] between the GORE approaches KAOS and iStar. While KAOS supports formal specifications, iStar offers a richer representation of goals, softgoals (qualities), non-functional requirements, explicit treatment of actors, roles, and dependencies, as well as more flexible task modeling. For risk analysis, iStar enables the representation of obstacles and vulnerabilities via contribution types and dependencies, offering a strategic perspective aligned with organizational objectives. These advantages motivated our choice of iStar for iStar4Deception.

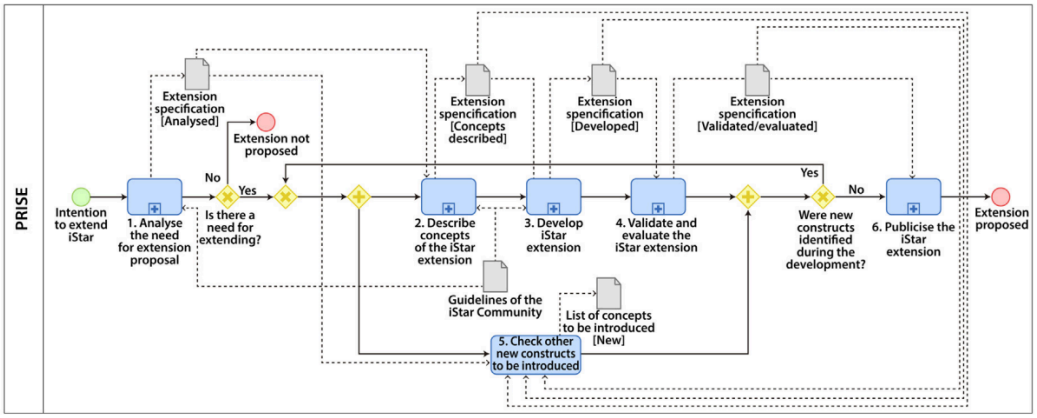

Figure 3. PRISE Main Process

To ensure conceptual rigor, interoperability, and semantic consistency, the development of iStar4Deception systematically followed the six phases of the PRISE methodology [Gonçalves, 2019] (Figure 3): (1) **Need Analysis:** At this phase, we identified through the literature review a conceptual gap in traditional security extensions (e.g., Secure Tropos, Elahi et al.), which lack intentional constructs for modeling active deception and adversarial manipulation; (2) **Concept Description:** At this phase, fundamental cyber deception concepts, including deceptive goals, tactics, operational techniques, and fake assets, were mapped into intentional requirements abstractions; (3) **Extension Development:** At this phase, the metamodel was constructed by reusing malicious constructs from Elahi et al. [Elahi et al., 2010] and introducing four new stereotyped metaclasses (*Deception Goal*, *Deception Tactic*, *Deception Technique*, and *Deception Resource*) fully integrated with iStar 2.0 and piStar-EXT; (4) **Validation:** At this phase, the extension was evaluated through a real-world Brazilian bank slip payment system case study (Section 6) and an ongoing structured empirical study with domain experts; (5) **Construct Refinement:** At this phase, feedback is collected from the case study and analyzed to ensure the proposed constructs

provide complete domain coverage without introducing redundant elements; and (6) **Publication:** At this phase, the metamodel and graphical notation are documented to foster reuse and integration within the Requirements Engineering community [Gonçalves, 2019]. Following this process, iStar4Deception systematically reused existing cybersecurity extensions before introducing new modeling elements. In particular, it incorporates the malicious constructs proposed by [Elahi et al., 2010], while introducing new metaclasses only for concepts that are unique to cyber deception. Adhering to PRISE also ensured compatibility with the iStar 2.0 metamodel and the piStar-EXT framework, facilitating the integration of cyber deception concepts into established goal-oriented requirements engineering practices [Gonçalves, 2019].

5 Results

The iStar4Deception extension was developed following the step-by-step process defined in the PRISE process [Gonçalves, 2019].

To ensure consistency with established cyber deception frameworks, iStar4Deception incorporates concepts from the cybersecurity literature and explicitly distinguishes strategic objectives from operational mechanisms [Han et al., 2018]. Following the iStar 2.0 guidelines [Dalpiaz et al., 2016], the extension introduces four dedicated constructs:

- Deception Goal: Represents the objective to be achieved through the use of deception strategies. It is represented by a goal with stereotype «Deception Goal».
- Deception Tactic: Represents the tactics used to carry out defense or deception actions in order to achieve the goal. A deception tactic may require multiple deception techniques to be successful. It is represented by a task with stereotype «Deception Tactic».
- Deception Technique: Represents the techniques used to ensure that the defense or deception tactics work effectively. It is represented by a task with stereotype «Deception Technique».
- Deception Resource: Represents a resource used to deceive the attacker, making them believe they are in a real and unmonitored system. These resources are designed to attract the attacker and allow their activities to be monitored. It is represented by a resource with stereotype «Deception Resource».

These constructs were developed as part of a security solution for a presence control system in Multi-paradigm deception modeling for cyber defense [de Faveri, 2021], in which deception mechanisms are deliberately embedded to enhance system security.

The structural organization and semantics of these constructs are formalized in the iStar4Deception metamodel (Figure 4). The metamodel integrates three class sets: original iStar 2.0 constructs (yellow) [Dalpiaz et al., 2016], reused security extensions from [Elahi et al., 2010] (blue), and the newly proposed deception metaclasses (green). Figure 5 illustrates both the new and reused graphical notation supported by piStar-EXT.

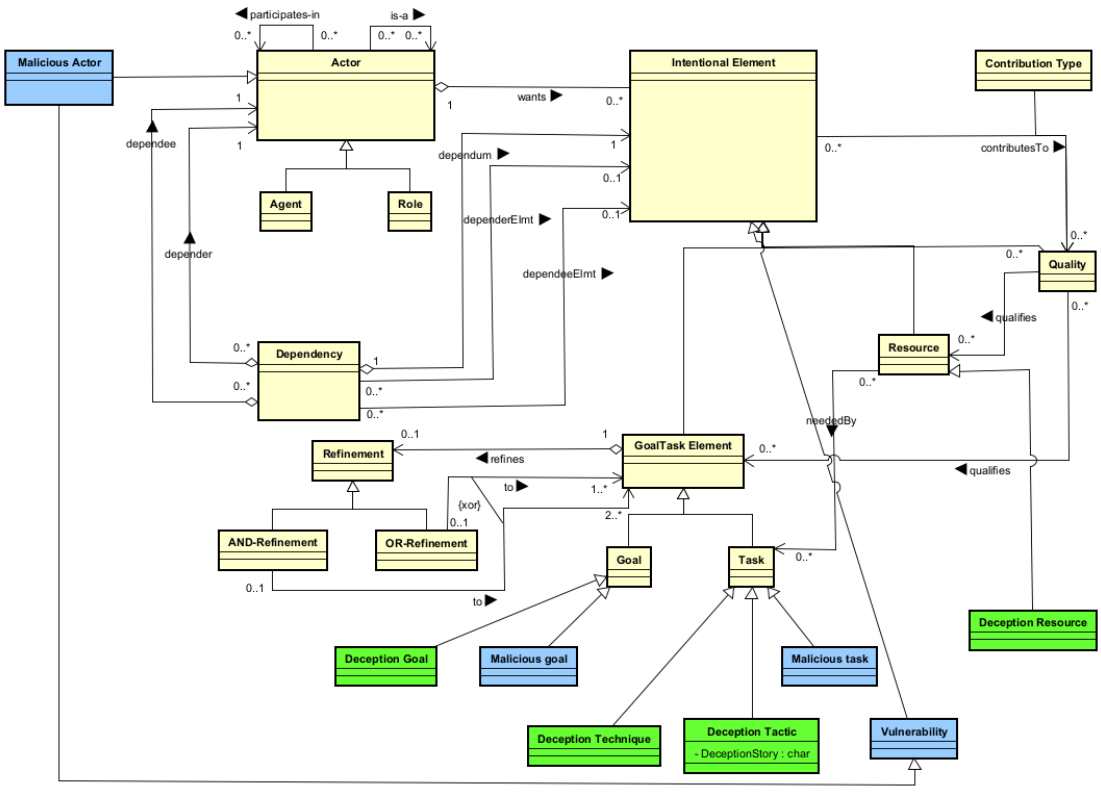

Figure 4. iStar4Deception Metamodel

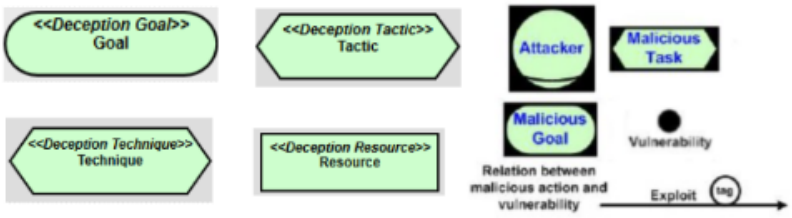

Figure 5. New and Reused Constructs [Elahi et al., 2010]

5.1 Discussion

The development of iStar4Deception addresses a critical gap in Goal-Oriented Requirements Engineering (GORE): representing cyber deception not as an isolated run-time control, but as an explicit, intentional strategy integrated into system architecture requirements [de Faveri, 2021].

By explicitly distinguishing high-level defensive objectives («Deception Goal») from operational behavior («Deception Tactic» and «Deception Technique») and physical or logical decoys («Deception Resource»), the extension provides two major benefits to security requirements engineering:

- Multi-level Abstraction: System architects can model deception strategies across different levels of abstraction [Han et al., 2018]. Early in the requirements phase, stakeholders reason about intentional goals (e.g., delaying attackers or gathering threat intelligence) [van Lamsweerde, 2001]. As architectural decisions mature,

these intent-level goals are systematically refined into concrete tactical mechanisms and deceptive assets [Dalpiaz et al., 2016].

- Trade-off and Vulnerability Analysis: Integrating malicious constructs (Attacker, Malicious Goal, Vulnerability) [Elahi et al., 2010] with deceptive constructs enables defenders to evaluate operational trade-offs. Requirements engineers can explicitly map how deception assets misdirect adversarial intentions away from critical production assets, preserving system integrity while capturing attack patterns in real time [Samhruth et al., 2025].

Furthermore, adhering to the PRISE process ensures that iStar4Deception maintains full semantic interoperability with native iStar 2.0 constructs and existing security extensions [Gonçalves, 2019]. This allows requirements models to capture traditional security mechanisms alongside active deception strategies within a unified notation.

6 Illustration

To demonstrate the applicability of iStar4Deception, we modeled a real-world scenario of a Brazilian Bank Slip (*boleto*) Payment System deployed by a large financial institution (Figure 6). The solution comprises an Internet Banking portal, public APIs for payment settlement via CNAB files, and a back-office administrative panel.

To implement cyber deception without modifying core business applications [Kahlhofer and Rass, 2024], deception mechanisms were integrated at the application and network layers. The system is deployed in a segmented VLAN-based intranet, while its web-facing components are hosted in a DMZ protected by a Next-Generation Firewall and an access control gateway. Automated routing controls isolate threats and redirect malicious interactions toward deceptive resources [Al-Sulaifanie et al., 2024, Nagahama et al., 2012].

Critical assets selected for deception coverage include payment settlement APIs, CNAB billing databases, administrative Single Sign-On (SSO) portals, internal message queues, and cryptographic HMAC signing keys. To mitigate unauthorized slip generation, parameter tampering, and data exfiltration, the architecture integrates an emulated deception network, automated host isolation, honeytokens embedded within critical assets, and Security Operations Center (SOC) capabilities, including Endpoint Detection and Response (EDR), Security Information and Event Management (SIEM), Security Orchestration, Automation, and Response (SOAR), and centralized logging.

The solution comprises seven main actors: *User*, *Attacker*, *DMZ*, *Intranet*, *Deception Network*, *Security Network*, and the *Back Office Analyst*. As illustrated in Figure 7, the *User* performs legitimate operations, whereas the *Attacker* attempts to compromise system assets. The *DMZ* forwards legitimate requests and redirects suspicious activities to the *Deception Network*, while notifying the *Security Network*. The *Deception Network* applies deception techniques to confine attackers in a controlled environment while collecting threat intelligence. The *Intranet* hosts core banking services and redirects suspicious activities to the *Deception Network*, protecting production assets. Meanwhile, the *Security Network* supports incident response by analyzing alerts, blocking malicious IPs, and protecting network assets, while the *Back Office Analyst* manages administrative workflows through workstation monitoring and automated traffic redirection. Overall, the architecture

integrates cyber deception across the DMZ, Intranet, and Deception Network to protect critical assets and support continuous threat intelligence collection.

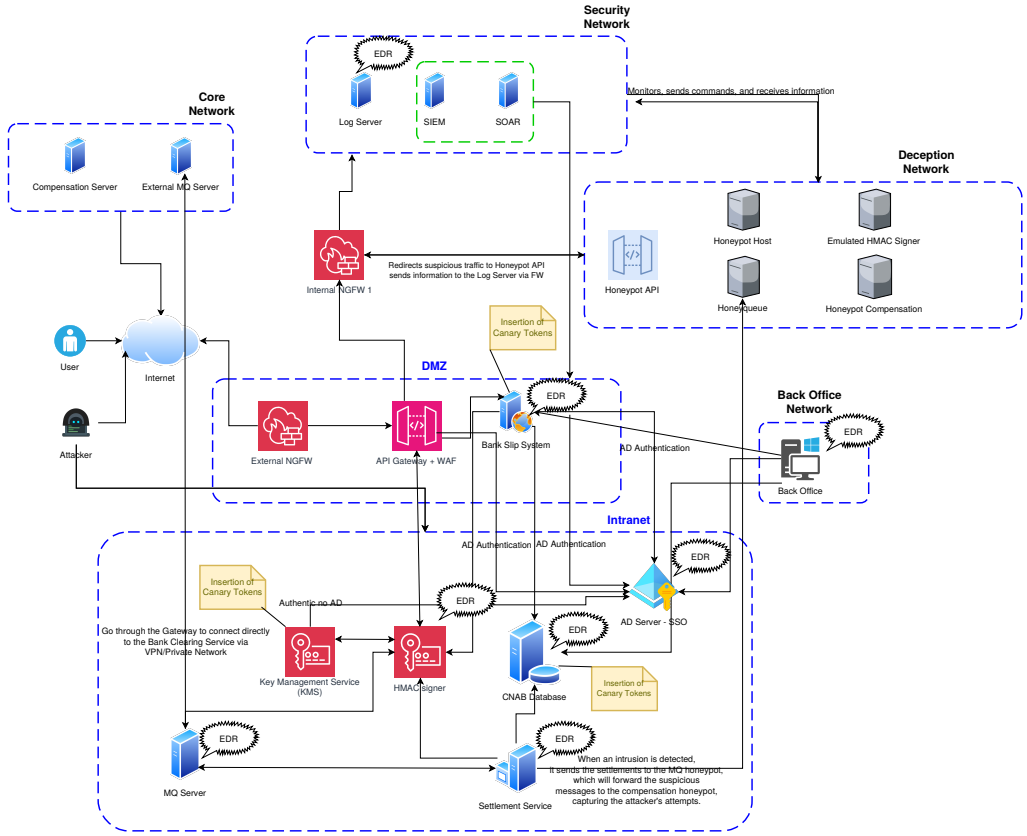

Figure 6. Solution Model

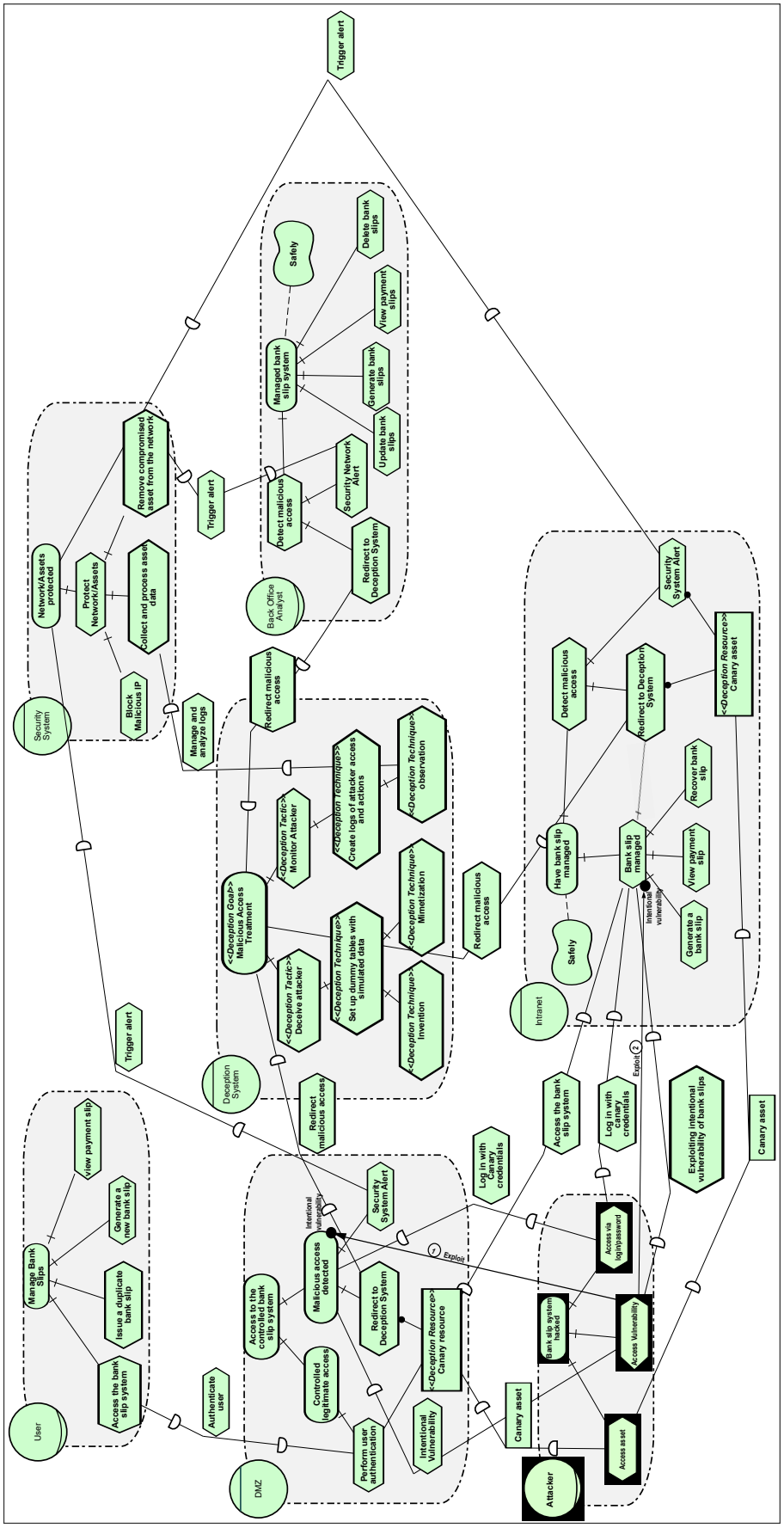

Figure 7. Solution Model iStar

7 Conclusions and Future Work

This paper presented iStar4Deception, a goal-oriented requirements engineering extension for systematically modeling cyber deception strategies alongside traditional security goals. Developed following the PRISE methodology, the extension introduces four dedicated metaclasses: Deception Goal, Deception Tactic, Deception Technique, and Deception Resource, while reusing existing vulnerability constructs to preserve consistency with the iStar 2.0 metamodel and the piStar-EXT framework. To demonstrate its applicability, we modeled a real-world Brazilian bank slip payment system, showing how cyber deception mechanisms and dynamic traffic redirection can be integrated into different architectural environments (DMZ, Intranet, and Deception Network) to protect critical assets, mislead attackers, and support threat monitoring. Overall, iStar4Deception extends goal-oriented requirements engineering with explicit support for cyber deception, enabling analysts to model proactive defense strategies from the earliest stages of system design. However, implementing cyber deception presents challenges, such as modeling realistic attack scenarios, maintaining deceptive data consistency, integrating with legacy systems, and avoiding impacts on legitimate users [Samhruth et al., 2025, Liebowitz et al., 2021]. As future work, and as an explicit step of the PRISE process (Phase 4: Validation and Evaluation), we are conducting a structured empirical evaluation with researchers and domain experts in Requirements Engineering and Cybersecurity. This study utilizes a quantitative and qualitative instrument to collect empirical evidence regarding the effectiveness, usability, comprehensibility, and perceived benefits of iStar4Deception compared to existing modeling approaches. Furthermore, we plan to apply the proposed extension to additional critical domains, such as healthcare and smart infrastructure, to evaluate its generalizability and expressiveness. This extension will enable the investigation of how security- and deception-related goals, dependencies, and softgoals can be systematically modeled across different contexts, in accordance with the principles of goal-oriented modeling in iStar [Yu, 1995, Mylopoulos et al., 1999]. The incorporation of automated mechanisms for generating iStar artifacts containing cyber deception elements directly from use cases and security policies is also envisaged.

In this context, based on the iStar model and taking deception extensions into account, we aim to develop a solution capable of producing a structured XML representation, supported by an XSD schema, with the purpose of formalizing model elements in a machine-readable format while preserving their semantics and dependencies, particularly those associated with security-related qualities and deception strategies [Mylopoulos et al., 1999]. This formalization enables syntactic and structural validation of the models and promotes their interoperability with tools supporting security engineering activities.

8 Acknowledgments

In this study, the generative artificial intelligence models Perplexity and ChatGPT were used to proofread and translate the text, tables and citations.

References

Al-Sulaifanie, A., Biswas, S., and Al-Anbagi, I. (2024). A survey of network requirements for enabling effective cyber deception. *IEEE Access*, 12:31450–31472.

Ceron, J. M., Steding-Jessen, K., and Hoepers, C. (2013). Anatomia de abusos a servidores sip. In *Anais do XIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)*, pages 44–57, Brasil. Sociedade Brasileira de Computação (SBC).

Climek, L. et al. (2016). Cyber deception in modern defense strategies. *Journal of the Cyber Security & Information Systems Information Analysis Center*, 4(1).

Cobalt Blog (2024). 11 biggest cybersecurity attacks in history. Online.

Dalpiaz, F., Franch, X., and Horkoff, J. (2016). istar 2.0: Language guide. *Requirements Engineering*.

de Faveri, C. (2021). *Modeling Deception for Cyber Security*. Phd thesis, NOVA University Lisbon, Lisbon.

Elahi, G., Yu, E., and Zannone, N. (2010). A vulnerability-centric requirements engineering framework: analyzing security attacks, countermeasures, and requirements based on vulnerabilities. *Requirements Engineering*, 15(1):41–62.

Franch, X., Maté, A., Trujillo, J. C., and Cares, C. (2018). On the joint use of i* with other modelling frameworks: A vision paper. In *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 33–36, Gothenburg, Sweden. ACM.

Giorgini, P., Rizzi, S., and Garzetti, M. (2005). Goal-oriented requirement analysis for data warehouse design. *8th ACM International Workshop on Data Warehousing and OLAP*, pages 47–56.

Gonçalves, E. J. T., Castro, J., Araújo, J., and Heineck, T. (2018). A systematic literature review of istar extensions. *Journal of Systems and Software*, 137:1–33.

Gonçalves, E. J. T. (2019). *PRISE: A Process to Support iStar Extensions*. Doctoral thesis, Universidade Federal de Pernambuco, Recife, Brazil.

Han, X., Kheir, N., and Balzarotti, D. (2018). Deception techniques in computer security: A research perspective. *ACM Computing Surveys (CSUR)*, 51(4):80:1–80:36.

Haseeb, J., Malik, S. u. R., Mansoori, M., and Welch, I. (2022). Probabilistic modelling of deception-based security framework using markov decision process. *Computers & Security*, 115:102599.

Horkoff, J. and Yu, E. (2016). Analyzing strategic actor relationships: A goal-oriented approach. *Data & Knowledge Engineering*.

ISO/IEC (2022). Information security, cybersecurity and privacy protection — information security management systems — requirements.

Kahlhofer, M. and Rass, S. (2024). Application layer cyber deception without developer interaction. In *2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 326–337. IEEE.

Kotenko, I., Fedorchenko, E., Novikova, E., and Jha, A. (2023). Cyber attacker profiling for risk analysis based on machine learning. *Sensors*, 23(4):2028.

Kotonya, G. and Sommerville, I. (1998). *Requirements Engineering*. John Wiley & Sons.

Lapouchnian, A. (2005). Goal-oriented requirements engineering: An overview of the research. Technical report, University of Toronto.

Liebowitz, D., Nepal, S., Moore, K., Christopher, C. J., Kanhere, S. S., Nguyen, D., Timmer, R. C., Longland, M., and Rathakumar, K. (2021). Deception for cyber defence: Challenges and opportunities. In *3rd IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications*.

Mitnick, K. and Simon, W. L. (2002). *The Art of Deception: Controlling the Human Element of Security*. Wiley.

Morandini, M., Penserini, L., Perini, A., and Marchetto, A. (2017). Goal-oriented requirement analysis for data warehouse design. *Requirement Engineering*, pages 77–103.

Mouratidis, H., Giorgini, P., and Manson, G. (2007). Secure tropos: A security-oriented extension of the tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, 17.

Mylopoulos, J., Yu, E., Chung, L., and Nixon, B. (1999). Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Transactions on Software Engineering*, 18(6):483–497.

Nagahama, F. Y., Farias, F., Aguiar, E., Gaspary, L., Granville, L., Cerqueira, E., and Abelém, A. (2012). Ipsflow – uma proposta de ips distribuído para captura e bloqueio seletivo de tráfego malicioso em redes definidas por software. In *Anais do XII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg)*, pages 324–330, Brasil. Sociedade Brasileira de Computação (SBC).

NIST (1977). Guidelines for automatic data processing physical security and risk management. Technical report, National Bureau of Standards.

Pacheco, F. and Staino, D. (2025). Reinforcement of cyber deception strategies through simulated user behavior. In *Proceedings of Cyber Deception Strategies*. Springer.

Pirmez, M., Pirmez, L., da Costa Carmo, L. F. R., Delicato, F. C., Pires, P. F., and de Sousa, E. B. (2008). Prometheus: Um serviço de segurança adaptativa. In *Anais do XXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2008)*, pages 229–242, Porto Alegre, RS, Brasil. Sociedade Brasileira de Computação (SBC).

Ribeiro, M., Castro, J., and Pimentel, J. (2019). istar for safety-critical systems. In *Proceedings of the 12th International i* Workshop*, volume 2490 of *CEUR Workshop Proceedings*.

Samhruth, A., Girish, K., and Ganesh, N. (2025). Siren: Advancing cybersecurity through deception and adaptive analysis. In Arai, K., editor, *Intelligent Computing*, pages 506–519, Cham. Springer Nature.

Shinde, A., Doshi, P., and Setayeshfar, O. (2021). Cyber attack intent recognition and active deception using factored interactive POMDPs. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '21)*, pages 1200–1208, Richland, WA, USA. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS).

Smith, D. L. (2005). *Why We Lie: The Evolutionary Roots of Deception and the Unconscious Mind*. Taylor & Francis.

Sommerville, I. (2011). *Software Engineering*. Pearson, 9 edition.

Spitzner, L. (2003). *Honeypots: Tracking Hackers*. Addison-Wesley.

Steingartner, W., Galinec, D., and Kozina, A. (2021). Threat defense: Cyber deception approach and education for resilience in hybrid threats model. *Symmetry*, 13(4).

van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *Proceedings of the 5th IEEE International Symposium on Requirements Engineering*, pages 249–262. IEEE.

Werneck, V. M. B., Oliveira, A. d. P. A., and Leite, J. C. S. d. P. (2009). Comparing gore frameworks: i-star and kaos. In *12th Workshop on Requirements Engineering*.

Whaley, B. and Bell, J. A. (1991). *Cheating and Deception*. Transaction Publishers, New Brunswick.

Yu, E. (1995). *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, University of Toronto.