INFORMATICA, 2023, Vol. 34, No. 1, 121–146

DOI: https://doi.org/10.15388/23-INFOR510

Causal Knowledge Modelling for Agile Development of Enterprise Application Systems

Karolis NOREIKA, Saulius GUDAS*

Institute of Data Science and Digital Technologies, Vilnius University,
Akademijos str. 4, LT-08412 Vilnius, Lithuania
e-mail: karolis.noreika@mif.vu.lt, saulius.gudas@mif.vu.lt

Received: July 2022; accepted: February 2023

Abstract. Experience shows that Agile project management tools such as Atlassian Jira capture the state of EAS projects by relying solely on expert judgement that is not supported by any knowledge model. Therefore, the assessment of project content against strategic objectives and business domain features are not supported by any tool. This is one of the reasons why Agile project management still does not provide sufficient EAS project delivery results. In order to address this problem, the Enterprise Application Software (EAS) development using Agile project management is summarized in a conceptual model. The model highlights the knowledge used and indicates its nature (empirical or causal digitized). The modified Agile management process we have developed and described in previous works is based on causal knowledge models that supports EAS development and Agile management processes. The purpose of this article is to specify knowledge repository to ensure the Agile management solutions of an EAS project are aligned with strategic goals and business domain causality. It is worth noticing that strategic goals have been identified and specified as capabilities using some enterprise architecture framework (NAF, MODAF, ArchiMate, etc.). The novelty of the proposed method is incorporating the business domain causal knowledge modelling approach into the Agile project management process. The causal knowledge unit is considered as a Management Transaction (MT), which includes closed loop dependence of its components. The modified Agile activity hierarchy (theme, initiative, epic, user story) defines the required content of their mutual interactions. An important new results obtained are the conceptual model of causal knowledge base (KB) and specification of enhanced Agile management tool components: project management database and project state assesment knowledge base. Causal KB includes specification of causal knowledge unit (MT metamodel) and specifications of traditional and causal Agile hierarchy meta-models. These conceptual models define the causal knowledge components necessary to evaluate the state of Agile activities in the EAS development project using intelligent Agile project management tool.

Key words: Agile management method, causal modelling, management transaction, knowledge base.

*Corresponding author.

1. Introduction

The main competitive advantage of modern-day enterprises is the ability to innovate and use information technology to support their business: help to manage the ever-increasing quantity of information, deal with the complexity of business processes, external regulations, and support operational activities.

Enterprise application software (EAS) are complex information systems used in modern-day enterprises as they service more than one enterprise management activity. EAS development and its development project management is a complex process and requires a specific approach in order to maximize the value and minimize the cost of development. Furthermore, translating the specifics of business domain internal interactions, such as business processes, strategy policies, regulations and general best practices to the requirements for EAS development projects is not a trivial task.

Business domain modelling allows to save costs when business processes are represented in a virtual environment to ensure processes are optimized: bringing most value to the enterprise with least cost or effort. In other case, when there is no modelling performed, the EAS is developed through trial and error approach, where the whole system is developed only based on the knowledge of business experts. Given the scenario of such development, there would be no option to simulate business processes and to optimize it in a virtual environment without making changes to real-world activities and processes to make sure that EAS supports them. EAS development in such a way is theoretically possible, but is the least optimal and most costly. Another example of not optimal EAS development is when there are IT development teams distributed across different departments in the company with direct reporting to the specific department manager (non-IT). The flaw of this approach is the gap between IT strategy, governance and best practices and the way anchored IT teams work. In order to optimize EAS development, the distributed teams must have a way to align their practices so there would not be too many differences in coding standards, quality assurance practices, etc. or be anchored in the IT department with allocated resources to service the needs of different departments.

Enterprise application software engineering today is based on model-driven architecture (MDA) and model-driven development (MDD) practices. Model-driven development (MDD) of the Enterprise Application Software (EAS) heavily relies on the business domain experts knowledge when enterprise business process models are created to support software development. MDA provides guidelines to model abstraction hierarchy and model transformations aimed to use business domain knowledge (captured on the computation independent modelling (CIM) layer) for EAS project development, separating the system project (Platform independent model (PIM) layer) from its implementation on a specific platform (Platform specific model (PSM) layer and code) (OMG, 2022a). MDD relies on domain experts to translate contextual business domain knowledge into models, as EAS development begins with business domain analysis and requirements gathering, which are transformed to CIM layer models. MDD allows to increase quality of complex software because meaningful validations can be executed on the high-level (CIM and PIM) models. This acquired knowledge is the basis for the transformation of the created models

into project models (PIM and PSM layers) of the new application software system. It is obvious that the quality and validity of further project solutions depend on the adequacy of the initial knowledge about the business domain and the understanding of the internal causal interactions discovered there. The MDD methodology creates the possibility to generate project solutions using the UML specifications, therefore full compatibility of UML models is required.

However, this method has limitations in understanding the causal relationship of the business domain, revealing the information content and causes of the processes, as it uses empirical modelling. Real-world processes in the business domain are modelled from the perspective of external observation using BPMN, UML, SysML or some other notations (OMG, 2022b). Causality (regularities) of a definite type of real-world domain (i.e. an enterprise in our case) must be understood and any representation of domain (model, framework) must fit those causality characteristics. The relatively new Agile approach to Model Driven Development (AMDD) does not ensure satisfactory EAS project delivery results, as only every 3rd project is completed as successful (KPMG, AIPM, IPMA, 2019). The analysis of the interaction between Enterprise Application Software (EAS) development and Agile management environment was carried out in order to determine the sources of knowledge and their nature (empirical or causal) that are used in these processes.

Causal modelling is a necessary basis in the analysis and development of the physical or cyber-physical systems, cyber-biological systems, as well as in the creation of cyber-enterprise systems. Understanding the behaviour of an enterprise requires a deep knowledge of the internal interactions of processes, so the study and discovery of business domain causality is a prerequisite. Other types of systems, i.e. for machine learning are also using causal analysis (Li *et al.*, 2022). Causal knowledge consists of the essential causal dependencies, which are inherent to the subject domain according to some theory or methodology (Gudas *et al.*, 2019). Our approach to Agile development management for EAS is based on the causal knowledge model named management transaction (MT), and the theoretical underpinning presented in Gudas (2012), Gudas and Lopata (2016). Sufficient control over the EAS design process is necessary to manage the complex Agile interaction efforts.

Researches (KPMG, AIPM, IPMA, 2019; digital.ai, 2020) prove the advantages of Agile methods over traditional project management methods. As the EAS development management process is complex, traditional project management methods like waterfall are falling short to support the complexity. As the obtained causal knowledge will eventually be implemented to a software solution, Agile methods are used for software development management. Agile methods help to deal with the complexity of EAS development by enabling transparency via inspection and adaptation to the software development process.

Although Agile methods do not define a specific way to ensure the decomposition of EAS project requirements, it is a general practice to capture the business requirements using a "user story". User stories provide an abstract but detailed enough description of the business problem. Nevertheless, the user story on its own is not enough to make sure that it will help to fulfill a business goal of a higher level, and additional levels of requirements are used. These Agile activities all together are called "TIES" standing for themes, initiatives, epics, and user stories (Prior, 2022). As one of the ways to deal with the complexity

of EAS project requirements specifications, using the Agile activities in the TIES format allows to link the Agile hierarchy elements and helps to manage complexity. Furthermore, if the activity on the lower level would not be completed (i.e. user story), the activity on the higher level (i.e. epic) would not be completed as well. This forms a cause-and-effect relationship. Despite that, the links between TIES structure elements are currently defined by human interaction and communication between project managers, Agile leaders, and business managers. This is where we believe the information about the deep causal knowledge is lost due to the lack of formalization of patterns, laws, and regularities of the real world – i.e. why a specific set of user stories are linked to a specific epic, a set of epics is linked to a specific initiative and a set of initiatives form a specific theme. This means that an additional definition of coordination between business and IT management is required to ensure business and IT alignment. This is not resolved by following the traditional requirements traceability approach (Gotel and Finkelstein, 1994). Although requirements traceability provides some structured way to map the requirements to business needs and project objectives, it still does not capture the causal knowledge of the domain. Discovering causal knowledge is essential to manage the complexity of business strategy and IT alignment and to create intelligent systems.

The modified Agile management model we have developed and described in detail in our previous works (Gudas and Noreika, 2022) comprises of causal knowledge models that influences EAS development and Agile management processes.

The purpose of this article is to examine the EAS engineering process, the sources of knowledge used there to formulate the requirements for the structure of the knowledge base of project management, based on the activity causal modelling method, and to specify the conceptual model of the knowledge base of the Agile project management system.

The remainder of the paper is structured as follows: Section 2 explains principles of MDA approach and explains the additional steps in the MDA/MDD process. Also, Agile Model Driven Development (AMDD) and causal modelling driven MDA/MDD approaches are discussed. Section 3 explains the causal Agile management process and contains the verification of developed models. Enhanced Agile project management tool architecture and Agile Development Management system knowledge base are described in Section 4. Section 5 presents conclusions that summarize the features of the developed conceptual model of causal knowledge base and the specification developed for the causal Agile management repository.

2. Related Works

2.1. *Transformations in MDA/MDD*

Model-Driven Architecture (MDA) is a software development approach by the OMG. The MDA approach includes the modelling of these layers: Computation Independent Model (CIM), Platform Independent Model (PIM), Platform Specific Model (PSM), and model transformations (inside each layer and between these layers) (OMG, 2022a).

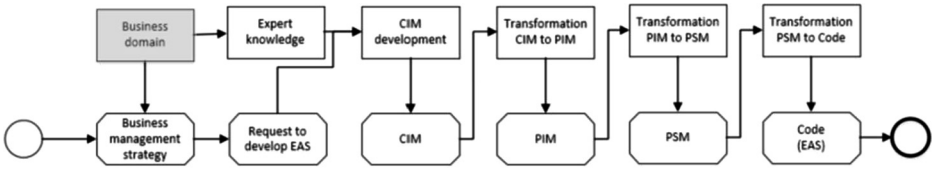


Fig. 1. Conceptual diagram of the MDA/MDD approach.

Traditional MDD is a model transformation process aimed to generate (semi automatically) the target model from the source model (Kleppe *et al.*, 2003). One of the shortcomings of the current MDD methods is that the initial knowledge about the business domain processes is described using empirical methods and models, relying on external observation and experience. Current MDD methods are not focused to understand the causes of internal interactions, to find the so-called deep knowledge. As stated in Gudas and Valatavičius (2020), "real world domain modelling at CIM layer is constructing of black boxes", therefore including causal modelling is a must from a theoretical point of view.

The conceptual diagram in Fig. 1 highlights the sources of knowledge that influence decisions in the MDA/MDD process – i.e. "Business management strategy", "Expert knowledge" about the "Business domain" and the "Request to develop EAS". These elements depict knowledge (preconditions) required for starting the MDA/MDD process of creating models in the CIM layer aligned with the business strategy.

Computation Independent Model (CIM) is designed to represent business domain processes (e.g. using BPMN and DMN notations, which are an OMG standard, OMG, 2022b), Platform Independent Model (PIM) specifies the EAS design model without regard to the hosting platform. Platform Specific Model (PSM) is the EAS project implementation model and includes concepts of the hosting platforms. Model transformations between the CIM, PIM, and PSM (inside each layer and between these layers) are defined.

MDD is a promising methodology for the development of cyber-physical-social systems (CPSS), cyber-enterprise systems (CES), cyber-physical systems (CPS), and other types of complex systems. Kulkarni *et al.* over the years have presented several papers aiming to improve the development of their component-based development environment Mastercraft (Kulkarni and Reddy, 2004). Mastercraft is a model-driven development toolset for developing large business-critical applications. It contains a meta-modelling tool to specify an abstract view. This represents the capture of predefined knowledge (i.e. grey-box approach). The MDA/MDD lies in one of the components of the Mastercraft tool – i.e. a set of code generators that transform each view instance to the desired implementation artifacts. Barat *et al.* has presented a configurable code generator meta-model to transform models to code (Barat and Kulkarni, 2010). Although some information that is required to smoothly conduct transformations are missing (i.e. "various non-functional concerns, namely, design strategies, technology platform, and architecture") the model described in Kulkarni and Reddy (2004) is well defined to be used for the MDD approach. The model transformation rules specify the mapping between MDA layers and lead to the specification of the computer code for implementation of the solution. The transformations between CIM, PIM, and PSM layers must comply with relevant meta-models (den Haan, 2008).

To summarize, current MDA/MDD methods do not contain the element of domain causality modelling.

2.2. *Agile MDA/MDD Methodology*

Agile Model Driven Development (AMDD) is an attempt to use the benefits of the fast-paced and responsive to change Agile development and the quality-focused and stable structures of Model-Driven Development. Ambler coined the term Agile Model Driven Development (Ambler, 2002). By examining the blended concepts of Agile and MDD it should be noted that MDD is a model-centric approach and values the thoroughness of models which take time for development. Agile, on the other hand, values working software, and the attention to models is little to none. Using the Agile approach, extensive models are considered as not required, and as Ambler states even with the AMDD approach, "an agile model is a model that is just barely good enough" (Ambler, 2004). Agile models need to exhibit a set of traits like fulfilling purpose, being sufficiently detailed, etc. However, it is not explained how the knowledge is obtained and there is no mentioning of performing model transformations as required by the MDD approach. This shows that model transformations are left to the MDD side of the AMDD approach. Furthermore, Ambler states that in order to perform modelling, fundamental information gathering skills are required and names observation as one of them (Ambler, 2004). This leads to the conclusion that Agile modelling is external observation-based (black-box). Ambler also stated that "AMDD is <only> about modelling MDD models more effectively" (Ambler, 2004).

Kulkarni *et al.* introduced a "meta sprint" based software development approach. Results show that there was a decrease in turnaround time from requirement specification to delivery, the increase of productivity of the development team, and less rework by using their developed tool Mastercraft (Kulkarni *et al.*, 2011). However, this approach is very similar to the dual Scrum or dual Agile approach (Miller, 2005) which later evolved into the so-called "design sprint" (Knapp, 2016). The dual-track Agile and design sprint was inspired by the "double diamond" methodology (British Design Council, 2007) formulated by the British Design Council. Furthermore, the approach by Kulkarni *et al.* does not describe the causal knowledge that the meta-models should explain.

Matinnejad (2011) has presented criteria how to evaluate the attributes of the fast-paced and flexible approach of Agile and thorough, well-thought, and complete approach of MDD and analysed the AMDD processes of Sage (Kirby, 2006), Hybrid MDD (Guta *et al.*, 2009), MDD-SLAP (Zhang and Patel, 2011) and the AMDD high-level life cycle (Ambler, 2004).

To summarize, after a thorough research to the best of our knowledge, there is no attempt to add the component of intelligence (causal knowledge capture) to Agile MDD methods.

2.3. *Causal Modelling Driven MDA/MDD Approach*

The traditional MDA/MDD method and AMDD method start with the analysis and modelling of the business area using the standard OMG notation BPMN, which allows to con-

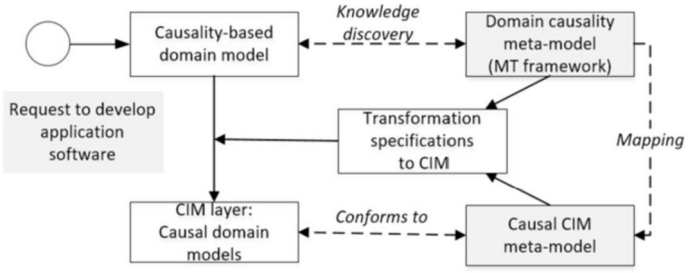


Fig. 2. The causal MDA/MDD transformations. Based on Gudas and Valatavičius (2020).

struct models from (Input, Process, Output) elements, connect them, create hierarchies (sub-processes). It is essentially an empirical modelling approach that creates black-box models without revealing the causality of the business domain. More advanced is DMN (Decision model and notation), which uses decision tables and is good at specifying complex business rules (OMG, 2022b). DMN models partially reveal causality dependencies and can be classified as a grey box type. Therefore, both the traditional MDA/MDD approach and the AMDD approach does not contain the domain causality aspect, therefore it is missing the key feature of creating an intelligent software development management system.

In order to improve this, we will utilize the causal modelling approach. Specific domain causality awareness is the prerequisite for discovering deep knowledge (i.e. regularities, laws) in a given domain. Causation methods are common in statistics, econometrics, cybernetics, computer science, data science, and other complex sciences to study cause-effect relationships and construct causal models in order to predict and control the possible dynamics of the systems (Gudas, 2021). From the causal modelling viewpoint, an enterprise is a subject domain, considered as a self-managed system driven by internal needs (strategy, management, system goals).

Based on causal modelling viewpoint the MDA/MDD approach was enhanced with a new layer (above CIM layer) of the domain knowledge discovery in Gudas and Valatavičius (2020) and the causal knowledge discovery (CKD) technique tailored for the enterprise domain. The peculiarities of the causal MDA/MDD method is illustrated in Fig. 2. In this block diagram rectangle means component, arrow – interaction.

We focus only on the stage of building a causal domain model and transforming it into a CIM layer, as these key steps apply to our approach to EAS development management using a modified Agile management hierarchy. As in this article we investigate the content of project management processes, in Fig. 1 the "Business domain" means project management domain (processes).

Causal domain models in the CIM layer are built using transformation specification rules, which defines mapping of the domain causality meta-model to causal CIM meta-model, and both meta-models are relevant to MT framework (predefined causal knowledge) (Gudas, 2021; Noreika and Gudas, 2021). The same principle is applied in the modified Agile management process (Noreika and Gudas, 2021), when predefined causal

Table 1
Analysis of causal perspective in the AMDD methods.

MDA transformation specification type	Domain model	Domain Meta-model	Domain knowledge type/modelling approach		
			Observation-based/ Black-box	Rule-based/ Grey-box	Causal knowledge/ White-box
Meta-model	Yes	Yes	–	Kulkarni and Reddy (2004)	–
Meta-model	–	Yes	Kirby (2006), Lano *et al.* (2015)	–	–
DSL	–	–	Grigera *et al.* (2012), Nakicenovic (2012)	Robles Luna *et al.* (2009)	–
Meta-model	Yes	–	–	Rivero *et al.* (2013), Rivero *et al.* (2014), Cáceres *et al.* (2004)	–
Meta-model	*Causal*	*Causal*	–	–	*Gudas and Lopata (2016), Gudas and Valatavičius (2020), Gudas (2021)*

knowledge (meta-model of Agile management hierarchy) is used to verify the status of EAS project solutions (project content) which is recorded in the standard Jira tool.

Alfraihi *et al.* have conducted a thorough systematic literature review on the topic of the integration of Agile development and MDA/MDD transformations (Alfraihi and Lano, 2017). We examined the results to determine if there is a description of how to capture causal domain knowledge. The conclusions are presented in Table 1 and are based on the causal modelling approach as described in Gudas and Valatavičius (2020), Noreika and Gudas (2021). The papers that do not have MDA transformations specified are not included. The last line in italic indicates a gap in the established MDD/MDA transformation approaches where the MDD/MDA methods are converging to a causal knowledge based transformation rules. "DSL" stands for "Domain specific language". Three types of knowledge are shown in Table 1:

- Observation–based knowledge (external modelling) corresponds to black-box approach, when the model elements are black boxes of the input-process-output type and their sequence does not have to form a feedback loop.
- Rule-based knowledge corresponds to a grey-box approach where deeper knowledge (fragments of causality) is known, but not yet sufficiently captured (external and internal modelling are mixed).
- Causal knowledge corresponds to the white-box approach when the domain causal model is known. In our case this is defined as the MT framework (Gudas, 2012; Gudas and Lopata, 2016; Gudas and Valatavičius, 2020).

Many tools support the Agile software project management process. Atlassian Jira software is one of the leaders (digital.ai, 2020). Current state of Jira only supports the themes, initiatives, epics, and user stories or similar structure. It does not provide formalities to ensure that the links between the different activities in the levels of the Agile

hierarchy are properly identified and justified in the business context. It means that it is not ensured that each and every user story, epic, initiative, and theme links to one of the strategic business objectives. Links are determined by experts working on project delivery and are subjective, which means that the links definition are prone to errors.

Furthermore, as it was presented in Table 1, only a few AMDD methods contain causality modelling, including meta-model, and in those articles there is a lack of definition of causal-based knowledge base or a similar repository, therefore a more thorough and detailed research is required.

3. Conceptual Model of EAS Development Management

The principles of the modified (causal) Agile process are as follows:

a) Agile hierarchy activities (theme, initiative, epic, user story) are implemented as management transactions, i.e. internal structure and interactions of Agile activities are corresponding to the definition of MT in Fig. 4;
b) Vertical interactions between levels of Agile hierarchy (theme, initiative, epic, user story) are implemented as management transactions corresponding to the definition of MT in Fig. 4;
c) Top level Agile activities themes are related to the strategic objectives of the enterprise, represented as capabilities specification in Fig. 6.

The conceptual scheme in Fig. 3 consists of two parts and shows an interface of traditional MDA/MDD development (on the left) and the modified Agile development management (on the right), based on the causal modelling. This conceptual diagram follows the notation of the MODAF Operational Node Relationship View OV-2 (British Ministry of Defence, 2010). A node here denotes a unit (hardware, software, organizational units, people, etc.) required to implement a specific EAS development capability (functionality). A node also expresses information flows requirements between nodes (British Ministry of Defence, 2010). A description of the nodes is presented in Table 2.

The conceptual model in Fig. 3 is divided into two different perspectives: the real-world (nodes M1, M3, M4, M6, and M7), and the virtual world (nodes M2, M5, and M7*). The link between the parts of the "Virtual World" and "Real World" as depicted by the relations of M2-M3, M2-M1 and M5-M3 illustrates the methodology of engineering – i.e. how the knowledge from the real world is translated to the requirements for EAS – in our case – the MDA/MDD approach. Content of M2 and M5 based on partly discovered M4 (through the experience of M1 and M3).

The brief characteristics of the nodes, defined in Table 2 are as follows:

- M1 – Experience-based business management processes (activities and responsibilities), related to the enterprise strategy requirements, operational goals. Corresponds to management functions (support activities of Porter's Value Chain Model).
- M2 – the virtual business process models, specified in some standard modelling notation (BPMN, UML, MODAF, etc.);

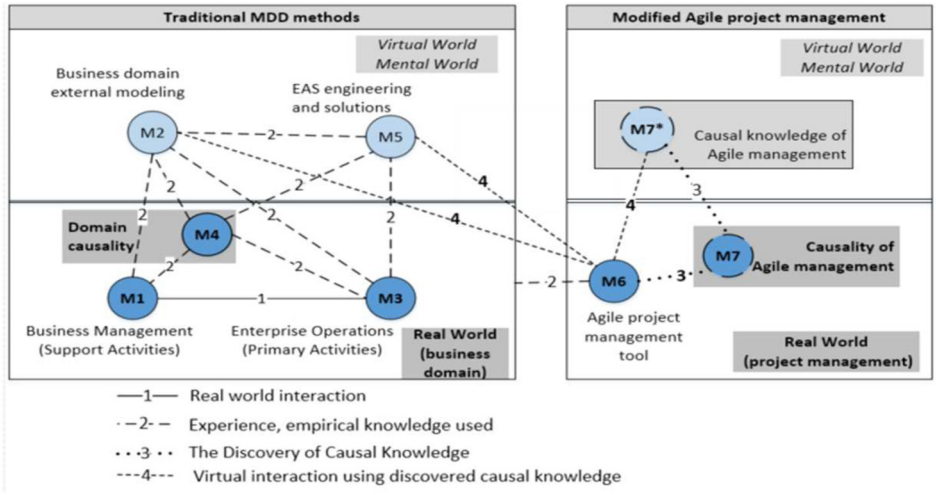


Fig. 3. Conceptual model of causal EAS development management.

Table 2
Causal Agile Development Management model nodes description.

Node ID	Name of node	Description
M1	Business management activities	Real-world processes (support activities of Porter's VCM, Porter, 1985)
M2	Business domain external modelling	Virtual models M2 describe the M3 and M1 activities
M3	Enterprise operations	Real-world processes (primary activities of Porter's VCM, Porter, 1985)
M4	Domain causality	Real-world causation (regularity) that is inherent for domain type
M5	EAS solutions	EAS development activities (development, testing, deployment)
M6	Agile project management tool	Computerized tool, supporting the enterprise strategy alignment with EAS solutions
M7	Causality of Agile management	Real-world activities, causal interactions between Agile activities (theme, initiative, epic, user story)
M7*	Causal knowledge base of Agile management	Based on the causal models: modified Agile hierarchy, and the management transaction (MT) framework

- M3 – Real-world processes (primary activities of Porter's Value Chain Model), i.e. physical processes (manufacturing, development) of products or services. Controls from M1 have impact to M3;
- M4 – Real-world regularities (patterns) that could be known to a certain extent as causal knowledge (deep knowledge). Causal dependencies of M1 and M3;
- M5 – EAS solution including project solutions and code, based on empirical models (M2) and experience related to M3 and M1;
- M6 – Intelligent Agile management environment, based on the causal Agile management model, focused on the enterprise strategy alignment with EAS solutions. Contains the recorded state of the EAS development (scope, status of completed features), including validation of project state against the causal model of Agile management interactions;

- M7 – The inherent interactions between items in the Agile hierarchy (theme, initiative, epic, user story);
- M7* – Causal knowledge constructs related to Agile management, i.e. causal knowledge base consisting of:

  a) meta-model of the modified Agile management hierarchy;
  b) project state (data) mapping to modified Agile management hierarchy, i.e. project state evaluation (verification) records.

Causal knowledge base specifies the causal knowledge as meta-models: management transaction (MT) meta-model (Gudas, 2021), modified Agile hierarchy and project state evaluation metrics (Gudas and Noreika, 2022).

We will examine traditional EAS development (Fig. 3 left side) and its relationship with causal Agile management processes (Fig. 3 right side) in order to describe the knowledge structures required to implement such a system.

3.1. *Model of the Traditional EAS Development*

The traditional EAS development part in Fig. 3 includes real world interactions between business management and operations (nodes M1–M3), and experience–based interactions between nodes (M1–M4–M2), (M3–M4–M2), (M3–M4–M5) and (M2–M5) that are explained in this subsection below. The interactions between nodes (M1–M4) and (M3–M4) shows the causality understood through experience – i.e. performing the real world activities multiple times reveals the peculiarities of the business domain and allows to capture the causality. The interaction between nodes (M4–M2), (M4–M5) means that the expert (business analyst or developer) must understand the business domain, i.e. understand the causality of it.

The traditional MDA methodology relies on the external business domain analysis of activities (M1–M3) and analyst's experience when the CIM layer models are constructed. There is no requirement that the domain causality M4 must be directly theoretically studied and used to develop M2 and M5.

The domain causality M4 exists as causality is a permanent feature of the real world and has impact on interactions between real world processes M1 and M3.

Business domain modelling is based on external observation, creating business domain models (M2) based on experience and using them to design EAS solutions M5. Content of M2 and M5 based on partly discovered knowledge M4 (through experience). However, this experience is not clearly expressed as a particular model(s) of domain causal knowledge.

The project management is considered as real-world process with its permanent causality (node M7) that needs to be understood thoroughly and modelled (M7*) to develop an intelligent project management system. Therefore, we model the Agile project management process as EAS engineering process following MDA approach. Thus, the Agile project management process as a causal process is described in Fig. 2.

The interactions between the aforementioned nodes are described as follows:

- The transitions (M1–M4–M2), (M3–M4–M2), (M3–M4–M5) imply that the causality of domain M4 has impact on modelling, i.e. creation of M2 and M5. However,

in traditional EAS development, this experience is not clearly expressed as a particular model(s) of domain causal knowledge;
- The transition M2–M5 means that business domain models M2 are translated (based on experience) to the requirements for EAS and directly used in the design of EAS solutions (node M5);
- It is important to note that usually M2 is specified in some standard modelling notation (BPMN, DMN, UML, etc.), which describes operational processes and data, but does not include the modelling of business strategic goals;
- Since the modelling of business strategic goals is a crucial moment in the modified Agile management process, we conclude that the content of node M2 needs to be supplemented by applying enterprise architecture frameworks that include strategy modelling specifications (e.g. MODAF, ArchiMate, NAF, etc.). The strategic goals are decomposed and finally represented by the "capability" concept defined in the before mentioned enterprise architecture frameworks.

Current MDA/MDD models on CIM layer are built through external observation, so the notations used for modelling do not distinguish between what is the control function (i.e. information transformation activity) and the controlled process (i.e. physical/material transformation activity). These notations do not require verification of the feedback loop, which must be created to ensure control causality.

It is not explicitly clear if there are only fragments of the whole system observed in node M2 or the whole components of the system and causal interactions are captured. In real-world EAS development projects this is often the case, because initially a lot of information about the processes are context-based, so the analyst or developer (external observer) cannot perceive the process in full, until it has sufficient knowledge of it and various exceptions, workarounds, etc. Only then proper modelling can be done.

We assume that business domain causality (M4) can be (partially) discovered and specified in the knowledge base. To achieve this goal, we apply the management transaction framework (MT), which is described in detail in previous works (Gudas and Noreika, 2022).

3.2. *Modified Agile Project Management*

Based on the analysis of the traditional Agile development management method, a modification is proposed in Gudas and Noreika (2022), Noreika and Gudas (2021) that is the conceptual background to create an intelligent EAS development management tool.

Conceptual model in Fig. 3 shows that causal Agile management tool M6 (right side of Fig. 3) is interacting with the traditional EAS development nodes (left side of Fig. 3) on the basis of experience and is supported with virtual transitions (M6–M2), (M6– M5) and (M6–M7–M7*).

Node M6 indicates the EAS project management tool (e.g. Jira, Azure Boards, Monday.com, Wrike, etc.), with a database about the current state of project execution (data) (state of project).

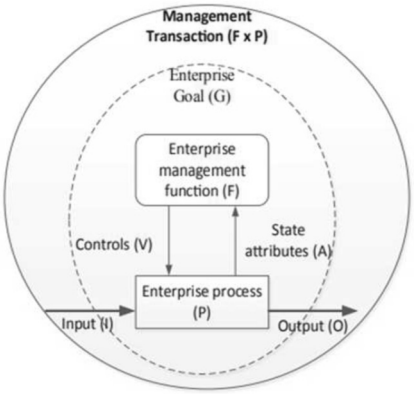


Fig. 4. Conceptual model of Management Transaction. Based on Gudas (2012).

M7 denotes EAS project management causality between Agile hierarchy of themes, initiatives, epics, user stories. These are real world activities, that we (partly) model using the meta-model of causal Agile management hierarchy (Fig. 6a)

Node M7* refers to project management knowledge base content described in Section 4: traditional Agile hierarchy meta-model, causal Agile hierarchy meta-model, and causal knowledge unit (MT meta-model).

The integration of M6 with M7* enables the creation of an intelligent Agile management environment, based on causal model of Agile management hierarchy (M7).

Due to these knowledge components, the node M7* enables to assess the project state specifications recorded in node M6 (project management tool database) and indicate the gaps against required causal Agile management hierarchy specifications (M7*). In our approach towards intelligent Agile management tools development, domain causation is the regularity of the enterprise management, defined as a management transaction (Gudas, 2012) and forms the basis of the knowledge base M7. Management transaction (MT) is considered as a causal knowledge unit. Recognized, discovered domain processes are specified by examining (verifying) whether they conform to the MT framework.

We provide further specifications for the knowledge base as part of node M7 and project management tool database additions as part of node M7 that allows to build an intelligent project management tool. The purpose of displaying M7 here as a component of the EAS development management is to demonstrate that we acknowledge that domain causality must be understood by the analysts, developers and any personnel working with EAS development to make sure that the developed EAS meets the real world processes.

The causal business domain model M4 is defined using the Management Transaction (MT) framework (Gudas, 2012; Gudas and Lopata, 2016). Management Transaction MT = (P, F, A, V) is relevant to some enterprise goal (G), captures knowledge on management function (F), enterprise process (P), informational input flow (state attributes A), informational output flow (controls V). MT includes a feedback loop between F and P composed of A and V flows.

A conceptual management transaction model presented in Fig. 4 also captures the enterprise goal (G), which is not specified explicitly in the formal MT definition.

Fig. 5. Agile management hierarchy (traditional), aggregation relationship.

Knowledge base (KB) is a crucial component of any intelligent system. KB contains the real-world knowledge captured by experts and is an internal model following the Internal model principle defined by Francis and Wonham (1976). The causal MDA/MDD approach uses predefined knowledge to ensure causality-based transformations (as depicted in Fig. 2). We believe that the KB needs to be created to support MDA transformations for developing EAS in an Agile setup.

3.3. *The Modified Agile Hierarchy*

Traditional Agile management hierarchy is based on the TIES hierarchy as explained in the Section 1 of this paper. Basically, concepts of themes, initiatives, epics and user stories represent the requirements for EAS development in different levels of abstraction. The mentioned concepts are activities – meaning actions are required to fulfil them – analysis, development, testing, etc. A set of initiatives form a theme, a set of epics form an initiative and a set of user stories form an epic. The traditional Agile management hierarchy is presented in Fig. 5. It must be noted that we do not investigate the link between strategic objective and theme in this paper, as this is a task of such complexity that it requires separate investigation. However, there are attempts to specify it using enterprise architecture frameworks (Noreika, 2021).

Analysis of traditional Agile hierarchy from the causal modelling perspective revealed a few new aspects to be considered in EAS project management as follows:

a) Different enterprise management functions are involved in EAS development (software development management and business development management);
b) There is an information flow between any two adjacent levels of the Agile hierarchy activities (themes – initiatives – epics – user stories);
c) The interaction of two activities from adjacent levels of the Agile hierarchy is a complex process with a feedback loop (there is circular causality).

By evaluating these characteristics of a real-world Agile process, we defined a modified (causal) Agile hierarchy model. This causal Agile management model is based on the business domain causality model and is focused on the enterprise strategy alignment with EAS solutions.

In the modified Agile hierarchy (Fig. 6a) any activity on each level:

- Is defined as management transaction MT = (P, F, A, V), i.e. themes, initiatives, epics, and user stories have a predefined semantical structure;
- Is considered as a white box in mutual interactions in the Agile hierarchy including horizontal and vertical interactions.

The detailed view of the modified Agile hierarchy is presented in Fig. 6a. An example of different activities in the Agile hierarchy layers is represented in Fig. 6b, here the axis

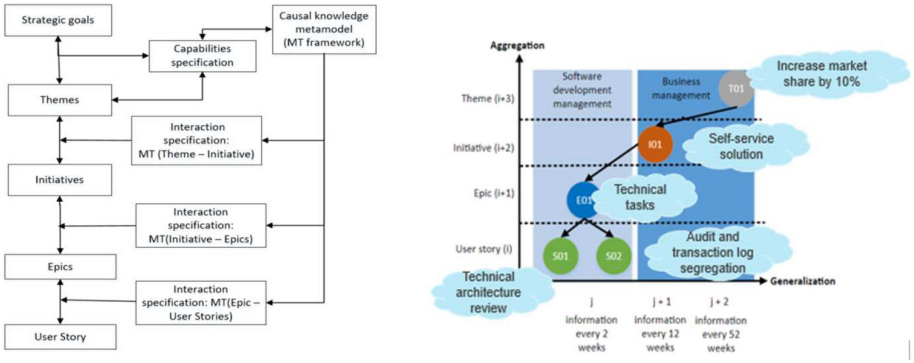


Fig. 6. Modified Agile hierarchy: a) detailed view; b) example of Agile activity hierarchy.

AG (aggregation) indicates the hierarchy of Agile activities (vertical interactions), the axis GE (generalization) indicates the classification of Agile activities into types. T01 in Fig. 6b represents example of themes in Fig. 6a, I01 – initiatives, E01 – epics and S01, S02 – user stories respectively. In our approach, the interaction of activities T01 and I01 is a management transaction that is a complex process and it is defined as a set of information flows specified in the element "Interaction specification: MT(Theme – Initiative) (Fig. 6a) and so on, respectively (I01 – E01, E01 – S01, E01 – S02).

Any vertical interaction between activities of different levels (theme, initiative, epic, user story, ...) is conceptualized as management transaction MT(P, F, A, V), where a higher level activity is considered as "management function" (role = F), relevant lower-level activities are considered as "processes" (role = P), also a feedback loop between higher-level activity (F) and lower-level activity (P) is predefined (state attributes flow A and controls flow V). Management Transaction is detailed in Fig. 4.

Vertical interactions of activities on the different Agile layers are overlapping. For example, the internal interaction in MT (initiative, epic) is as follows: the role of the higher-level element "initiative" is "control function" (F), and the role of the lower-level element "epic" is "process" (P). Next, the role of epic in internal interaction MT (epic, user story) changes: the role of epic here is F, the role of the user story is P.

A horizontal interaction between Agile activities must have a coordinating message between two management transactions MT(P, F, A, V) and MT*(P*, F*, A*, V*). Horizontal interactions differs depending on the management function of the managing function – i.e. software development management and business management.

3.4. *Verification of the Modified Agile Process*

The objective of model verification is determining if the implementation of the model is correct, i.e. conforms to the model requirement specification. The modified (causal) Agile process model must be verified to ensure that it meets causal modelling requirements defined as follows:

a) Agile hierarchy activities (theme, initiative, epic, user story) are implemented as management transactions, i.e. internal structure and interactions of Agile activities are corresponding to the definition of MT in Fig. 4;
b) Vertical interactions between levels of Agile hierarchy (theme, initiative, epic, user story) are implemented as management transactions corresponding to the definition of MT in Fig. 4;
c) The modified (causal) Agile process model ensure an interlink with enterprise strategy;
d) The knowledge repository structure ensures storage of internal elements of Agile activities defined as MTs in Fig. 4.

The verification rules a), b) and c) define requirements to conceptual model of the causal Agile management process and help to evaluate if the modified Agile process is performing the way it should following causal modelling paradigm.

The verification rule a) defines requirement that internal structure of Agile activities must include all MT elements (F, P, A, V). Verification rule b) defines the requirement that the internal structure of the interaction between the levels of the Agile hierarchy corresponds to the MT definition, i.e. would be a transaction with a feedback loop.

Verification rule c) defines the requirement that the Agile hierarchy must meet the strategic goals of the enterprise. Verification rule d) checks whether the causal elements of the Agile process, whose internal structure corresponds to the MT definition, can be stored in the designed project knowledge repository (Fig. 9).

The verification rule a) is satisfied because all activity types of causal Agile hierarchy (theme, initiative, epic, user story) are considered having internal structure defined as MT(P, F, A, V). For example, starting with the theme (Th), the internal structure is defined as MT(Th) = MT(P, F, A, V) as depicted in Fig. 8. The formal specifications of all causal Agile hierarchy activities, which are based on the MT definition, are discussed in detail in our article (Gudas and Noreika, 2022).

The verification rule b) is satisfied because causal model of each item of causal Agile hierarchy is defined as MT(P, F, A, V) (see rule a), thus by definition it includes vertical interaction between higher level and lower level activities.

In order to specify the content of MT more precisely, we include both identifiers of the higher level and lower level items in the specifications of MT structure. For example, in MT(Th,I) = MT(P, F, A, V), a lower-level activity "initiative" is considered as process (P) associated with a higher level activity "theme". The role of activity "theme" in this interaction is "management function (F)" (Gudas and Noreika, 2022). Therefore, we have the following specification: theme (Th) is defined as MT(Th, I) = MT(P, F, A, V) and includes a vertical interaction with initiative (I) which is defined as MT(I, E) = MT(P*, F*, A*, V*) (Gudas and Noreika, 2022) as illustrated in Fig. 8. The same structure of vertical interaction exists among all causal Agile hierarchy activities. The causal Agile hierarchy interaction specifications are discussed in detail in Gudas and Noreika (2022). Thus, the lower Agile activity satisfies the "needs for information" of the higher level activity – requirements for complete information about its state, i.e. a vertical interaction corresponds to required MT structure.

The verification rule c) is satisfied by the modified Agile process because top-level Agile activities themes are related to the strategic goals of the enterprise represented as

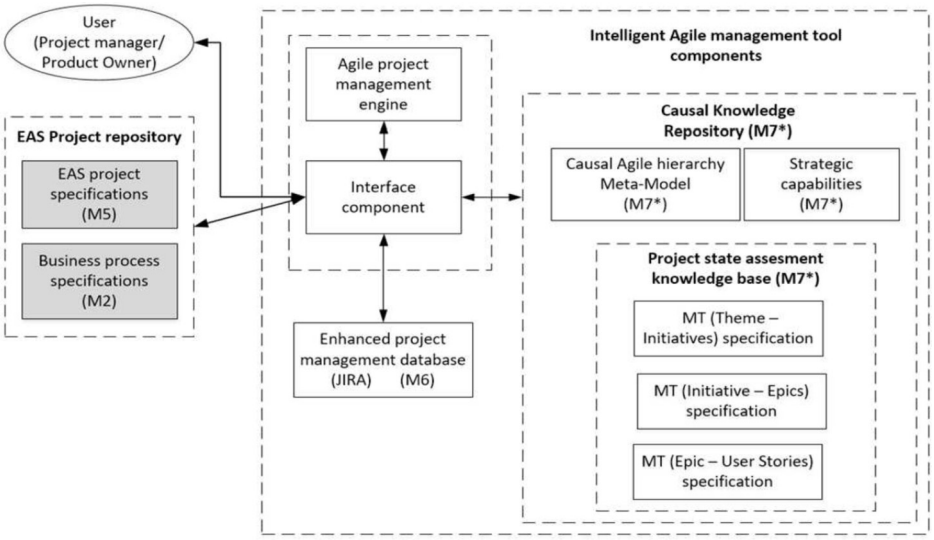


Fig. 7. Enhanced Agile project management tool architecture.

capabilities specification in Fig. 6 and attribute ID_CAPABILITY in the table Agile_activity in the knowledge repository (Fig. 9b). At the model implementation level, data storage tables (Agile_activity, MT in Fig. 9) have relationship with the data storage table (CAPABILITY).

Verification rule d) is satisfied because all Agile activities are saved as management transactions, because all internal MT elements (F, P, A, V) are specified in data storage tables (MT, A flow attributes, V flow attributes) in Fig. 9.

In addition, an example of the enhanced Jira tool database record in Table 3 demonstrates that the designed knowledge base meets the requirements of the causal Agile process.

4. Specification of the Knowledge Base for Agile Development Management

To develop EAS in an intelligent way and ensure that every task in the EAS project contributes to strategic business objectives and domain causality, using predefined knowledge is a must. By following the principles of modified Agile hierarchy the aim is to shift from the black-box or grey-box approaches to the white-box approach by capturing domain causal knowledge via MT framework. The architecture of the enhanced Agile project management tool that follows these principles is presented in Fig. 7. In this block diagram the ellipse means actor, dotted line – grouping, rectangle – component and arrow – interaction.

"User" in Fig. 7 represents the managers for the Agile EAS project. They are interacting with the enhanced Agile project management tool (i.e. Jira or any other that would be enhanced by the AI component).

The “Intelligent Agile project management tool” is represented by nodes M6–M7* in the conceptual model of EAS development management (Fig. 3). The intelligent Agile project management tool includes components as follows:

- “Agile project management engine” – regular features that Agile management tools have, i.e. backlog, sprint backlog, various attributes, reports, etc.;
- The “Interface component” is the user interface which ensures the real-time interaction with all of the components of the enhanced Agile project management tool. The “Interface component” interacts with the knowledge repository (node M7* in Fig. 3) and project management database (node M6 in Fig. 3);
- The “enhanced project management database” contains the information about TIES activities, like task descriptions, etc.;
- “Causal Knowledge Repository” includes the causal knowledge frameworks (meta-models), used for validation of the EAS development solutions: modified Agile hierarchy Meta-Model, capabilities specifications, MT-based specifications of vertical interactions in the Agile hierarchy and is represented by node M7* in Fig. 3. “Causal Knowledge Repository” consists of three interrelated parts: causal knowledge unit – the MT meta-model, traditional Agile hierarchy meta-model, and meta-model of the causal (modified) Agile hierarchy, including meta-models of the closed loop interactions between Agile hierarchy levels: meta-model of MT(theme, initiatives), meta-model of MT(initiative, epics) and meta-model of MT(epic, user stories). These interaction meta-models of the levels of the Agile hierarchy are considered subclasses of the unit of causal knowledge – the MT meta-model, and therefore, inherit upper level class features: they have function-MT, process-MT, Flow A and Flow V in their structure.

The “EAS project specifications” element under the “EAS Project repository” represents node M5 from Fig. 3. It contains EAS project specifications (in UML, SysML, etc.) that are used for EAS development.

The “Business process specifications” element under the “EAS project repository” represents the node M2 from Fig. 3. It contains the observation-based business domain models (in BPMN, DMN, etc.).

The system architecture in Fig. 7 is used as a roadmap to define the conceptual model (Fig. 8) and components of the causal knowledge repository (Fig. 9).

The conceptual model of the causal knowledge repository includes knowledge structures as follows (Fig. 8): a meta-model of traditional Agile hierarchy integrated with the capabilities specifications (relationship to strategic goals), causal (modified) Agile hierarchy meta-model and meta-model of causal knowledge unit (i.e. MT meta-model) which predefines required content of interlevel interactions that must match the MT definition.

Traditional Agile hierarchy meta-model defines the types of activities at different levels of the Agile hierarchy and the rule that the activity type “themes” is linked to the capabilities specification. This relationship ensures tracing of capability relationship to all hierarchy levels – from themes to initiatives, epics, and user stories.

This traditional Agile hierarchy meta-model supplemented by the connection with the capability is related to the Causal Agile hierarchy meta-model. The meta-model of causal

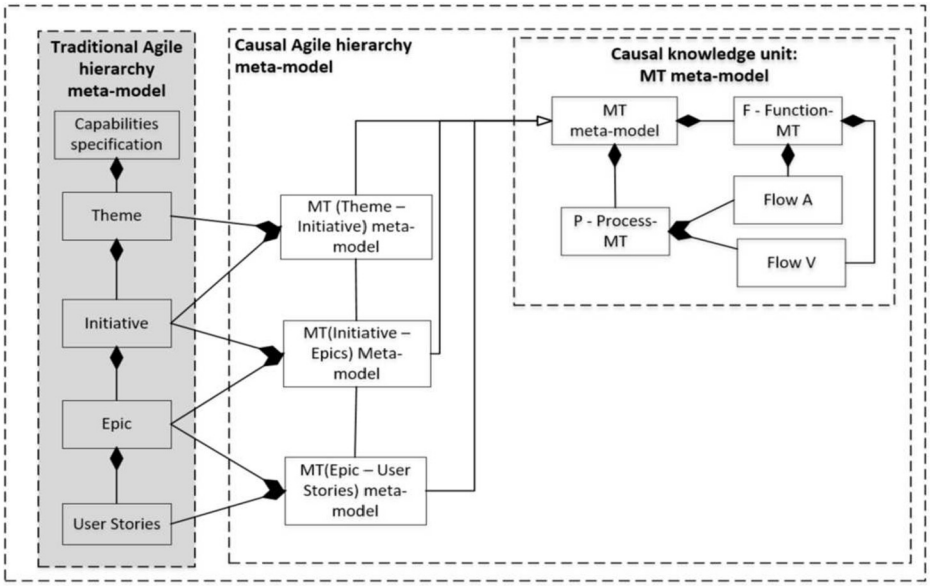


Fig. 8. Conceptual model of causal knowledge repository (UML Class diagram).

Agile hierarchy defines interlevel interactions of activity types (theme – initiative, initiative – epic, epic – user story). The internal structure (content) of these all vertical interactions is predefined by the Causal knowledge unit (here it is the MT meta-model).

Therefore, the Causal knowledge unit (MT meta-model) has a particular role in ensuring causality-driven vertical interactions: all interactions in the Agile hierarchy must match the MT definition: MT (theme, initiative) and MT(initiative, epic) and MT(epic, user,story) are defined as MT = (P, F, A, V).

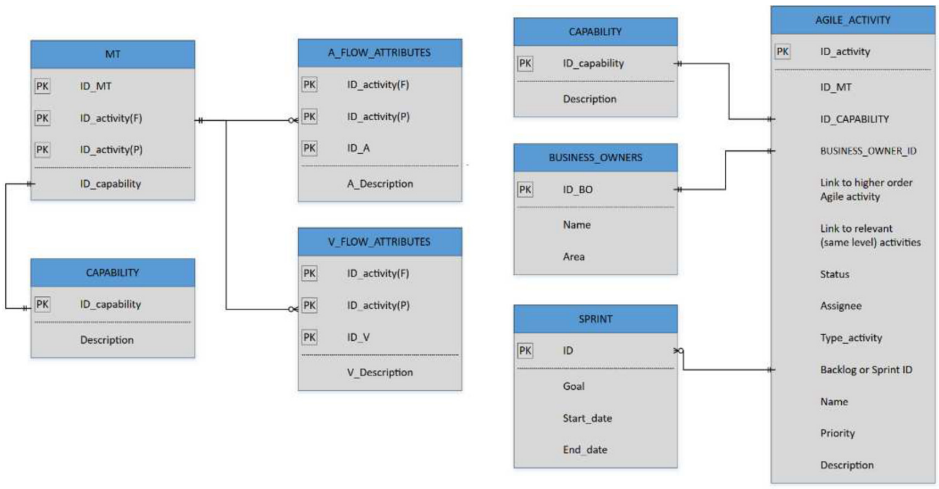
The causal Agile management meta-model that is used for validating the interlevel interactions is explained in more detail in Gudas and Noreika (2022).

The causal knowledge unit (MT meta-model) in Fig. 8 specifies requirements for the content of all interlevel interactions and is used by the intelligent system for validating the expert solutions (project state).

4.1. *Defining Agile Project Management Data Structures*

The causal Agile management repository specification (Fig. 9) is developed following the conceptual model of causal knowledge repository in Fig. 8. The causal Agile management repository in Fig. 9 consists in to two parts: enhanced project management database (Fig. 9b) and project state assement knowledge base (Fig. 9a).

Enhanced project management database (prototype is the Jira database) is supplemented with new DB tables CAPABILITY, BUSINESS_OWNER and new attributes required by the conceptual model in Fig. 8. The table CAPABILITY contains identifier (ID_capability) and description of strategic goal item. For ease of visualization,



a) EAS project state assessment knowledge base

b) Enhanced project management database

Fig. 9. Causal Agile management repository specification.

the CAPABILITY table is recreated twice. The table Agile_activity contains new attributes that ensure data storage of causal interactions: ID_MT – identifier of management transaction, ID_CAPABILITY – identifier of capability (i.e. strategic goal item), BUSINESS_OWNERS_ID – identifier of top level manager. The table Sprint is not modified.

EAS "Project state assessment knowledge base" in Fig. 9a is the implementation of the component "Causal Agile hierarchy meta-model" in Fig. 8. The causal knowledge storage structure includes tables MT, A_FLOW_ATTRIBUTES and V_FLOW_ATTRIBUTES with functional dependencies between them. This data structure ensures the storage of causal interactions content as defined in the conceptual model of causal knowledge repository (Fig. 8):

- Table MT ensures storage of MT = (P, F, A, V): identifier ID_MT, identifier of higher level activity ID_activity(F) (F – management function) and identifier of lower-level activity ID_activity(P) (P – process);
- Table A_FLOW_ATTRIBUTES ensure storage of MT = (P, F, A, V) flow A (information of process P state);
- Table V_FLOW_ATTRIBUTES ensure storage of MT = (P, F, A, V) flow V (decisions/controls of process P state).

It is important to notice that "Project state assement knowledge base" is linked to capability description in table CAPABILITY. This allowed for tracing of EAS solution activities (theme, initiative, epics, user stories) against strategic objectives.

According to the causal modelling method, project state data on interactions between levels of the Agile hierarchy are normalized by checking their compliance with the causal knowledge unit definition MT = (P, F, A, V). The storage of the received restructur-

Table 3
Content of the data base record of the enhanced Jira tool.

Theme	Initiative	Epic					User story				
		ID	F	P	A	V	ID	F	P	A	V
N	N	E01 Technical Tasks	N	Y	Y	Y	S01 Technical architecture review	Y	Y	Y	Y
N	N	E01 Technical Tasks	N	Y	Y	Y	S02 Segregate audit log and transaction log	Y	Y	Y	Y

ing result is ensured by the specifications of tables MT, A_FLOW_ATTRIBUTES and V_FLOW_ATTRIBUTES, it can be seen in Fig. 9a.

Thus, the specifications of the causal Agile management knowledge repository comply with the definition of the conceptual model of causal knowledge repository.

Fig 9b includes the implementation of the "Strategic capabilities" component of the "Causal knowledge repository" from Fig. 7.

In summary, the EAS "Project Status Assessment Knowledge Base" and "Enhanced Project Management Database" specifications (Fig. 9) meet the constraints defined by the conceptual models (Fig. 7 and Fig. 8).

The Fig. 9 shows how each causal Agile hierarchy interaction (MT(theme, initiative), MT(initiative, epic), MT(epic, user story)) should be defined and saves the actual state of current EAS project state assessment against strategic enterprise objectives.

Records of the Agile activities in the project management database are associated with a causal model of Agile hierarchy and interactions through an additional set of attributes (Fig. 9b), additional fields are capitalized.

We chose to present the fields with longer names for the sake of clarity. Furthermore, such usually encountered service information as date created, updated, user, that created or updated the record are not provided in each of the tables for the sake of simplicity.

Table 3 contains the records for EAS project requirements based on database structures in Fig. 9. It indicates that there is missing theme and initiative information and epic information is missing link to theme (F).

The prototype of an intelligent interaction component is presented in Fig. 10. This view is compiled using the content of the project management database record of the enhanced Jira tool from Table 3.

The dotted column in Fig. 10 reflects the possible additional fields based on the situation comparing records in the Agile project management database. Below is an example of the semantic information that the Jira tool would store (Jira database records content) and that is used to reflect the gaps of the project state against the modified Agile hierarchy meta-model:

- Theme represents a long-term objective, i.e. "in 2 years to reduce operational cost by X amount";
- One of the linked initiatives – "Stop using system Y, and transfer the features of the system to new system Z";

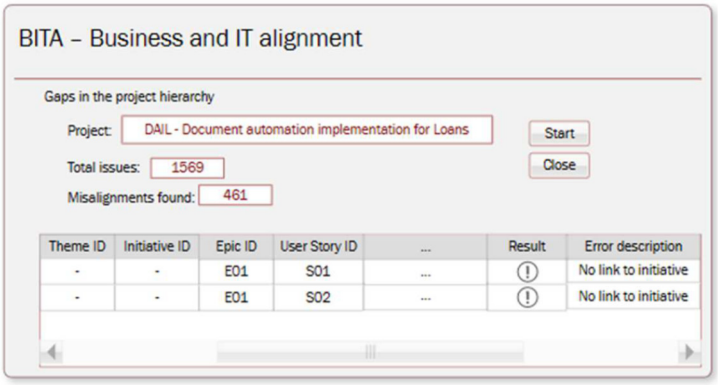


Fig. 10. Prototype of the enhanced Jira tool report.

- One of the linked epics to the initiative: "in 3 months – transfer system Y function N to system Z";
- One of the linked to the epic user stories: "in 2 weeks to transfer system Y function N component K to system Z".

5. Conclusions

EAS engineering methodologies lack the formalization to ensure the improvement of EAS project delivery. Furthermore, Agile project management tools, such as Atlassian Jira, capture the state of EAS projects by relying solely on expert judgement that is not supported by any knowledge model, causal knowledge is still not addressed in the MDD and AMDD approaches.

The novelty of the proposed method is incorporating the business domain causal knowledge modelling approach into the Agile project management process. The principles for development components for the intelligent Agile project management system that supports the application development project state evaluation were presented.

The presented conceptual model of causal EAS development management integrates causal knowledge into EAS design. Incorporating the causal models into the EAS development process and project management system enables the capability for validation of project content (i.e. specifications of user stories, epics, initiatives) using causal knowledge repository, including the alignment of strategic goals and EAS development solutions.

The basic component required for the intelligent Agile project management tool is the causal knowledge repository. A causal knowledge repository specification is proposed (based on the Jira tool). Repository consists of components as follows (required to store and verify causal knowledge): a traditional Agile hierarchy related to capability specifications, and a causal Agile hierarchy metamodel, including the specification of cross-level interactions. It also contains an Agile project management database of specific EAS project status that includes traditional Agile activity attributes and cause-based attributes.

This solution was verified by showing that the knowledge repository is built on a management transaction definition (causal unit of knowledge) that normalizes the internal structure of Agile activities and interactions.

All this together allows for the creation of an intelligent project management system, which allows to evaluate the content of EAS development tasks (i.e. user stories, epics, initiatives, themes) and align them with the requirements of strategic goals. The evaluation process should include procedures as follows:

a) Tracing of enterprise strategic goals integrity with Agile hierarchy activities (theme, initiative, epic, user story) using the ID_capability attribute and information from KB tables AGILE_ACTIVITY and CAPABILITY (Fig 9);
b) Data compliance verification of the interaction content between Agile hierarchy layers (theme/initiative, initiative/epic, epic/user story) to the causal structure defined as MT framework, using the information from KB tables MT, A_FLOW_ ATTRIBUTES and V_FLOW_ATTRIBUTES (Fig 9a).

The originality of this approach is the causal modelling solutions which ensure a more successful project deliveries, supported with enhanced Agile project management tool Jira having application development process evaluation capability. Standard Agile project state information is transformed by mapping it to causal Agile process model. Each interaction in the Agile hierarchy is mapped to causal knowledge structure defined as management transaction (MT). This makes it possible to identify the gaps of EAS as-is project data (stored in traditional Jira tool) against the required causal knowledge structure.

Despite significant advantages, the limitations of applying this method is the requirement to have the the organization to be at a particular process capability maturity level with the business processes to be defined and modelled, i.e. at least at level 3 according to CMMI (2022).

The presented conceptual models and structural solutions explain and specify the main components that would allow the creation of an intelligent Agile project management tool (an additional feature to the Jira tool). The next research steps are to create a prototype of the enhanced Agile project management system by integrating causal knowledge structures corresponding to the described structure of the project's knowledge repository to the standard Jira database.

References

Ambler, S. (2002). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. 1st ed., Wiley, USA.

Ambler, S. (2004). *The Object Primer*. 3rd ed., Cambridge University Press, USA.

Alfraihi, H., Lano, K. (2017). The integration of agile development and model driven development – a systematic literature review. In: *5th International Conference on Model-Driven Engineering and Software*, pp. 451–458. https://doi.org/10.5220/0006207004510458.

Barat, S., Kulkarni, V. (2010). Developing configurable extensible code generators for model-driven development approach. In: *SEKE 2010*, USA, pp. 577–582.

British Design Council (2007). 11 Lessons: Managing Design in Eleven Global Brands. https://www.designcouncil.org.uk/fileadmin/uploads/dc/Documents/ElevenLessons_Design_Council%2520%25282%2529.pdf. Last accessed 2022/02/20.

British Ministry of Defence (2010). MOD Architecture Framework (MODAF). https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/36757/20100602MODAFDownload12004.pdf. Last accessed 2022/10/15.

Cáceres, P., Díaz, F.J., Marcos, E. (2004). Integrating an agile process in a model driven architecture. In: *NFORMATIK 2004 – Informatik verbindet, Band 1, Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), Ulm, 20.–24*, pp. 265–270.

CMMI Institute (2022). CMMI. https://cmmiinstitute.com/cmm. Last accessed 2022/12/26.

den Haan, J. (2008). MDA and Model Transformation. http://www.theenterprisearchitect.eu/blog/2008/02/18/mda-and-model-transformation/. Last accessed 2022/03/10.

digital.ai Software, Inc. (2020). 14th Annual State of Agile Report. https://info.digital.ai/rs/981-LQX-968/images/SOA14.pdf. Last accessed 2023/01/15.

Francis, B.A., Wonham, W.M. (1976). The internal model principle of control theory. *Automatica*, 12(5), 457–465.

Gotel, O.C.Z., Finkelstein, C.W. (1994). An analysis of the requirements traceability problem. In: *Proceedings of IEEE International Conference on Requirements Engineering*. IEEE, USA, pp. 94–101. https://doi.org/10.1109/ICRE.1994.292398.

Grigera, J., Rivero, J.M., Robles Luna, E., Giacosa, F., Rossi, G. (2012). From requirements to web applications in an Agile model-driven approach. In: Brambilla, M., Tokuda, T., Tolksdorf, R. (Eds.), *ICWE 2012: Web Engineering*, *LNCS*, Vol. 7387. Springer, Berlin, Heidelberg, Germany, pp. 200–214. https://doi.org/10.1007/978-3-642-31753-8_15.

Gudas, S. (2012). *Foundations of the Information Systems' Engineering Theory*. 1st ed., Publishing House of Vilnius University, Lithuania.

Gudas, S. (2021). Causal modelling in enterprise architecture frameworks. *Informatica*, 32(2), 247–281.

Gudas, S., Lopata, A. (2016). Towards internal modelling of the information systems application domain. *Informatica*, 27(1), 1–29. https://doi.org/10.15388/Informatica.2016.74.

Gudas, S., Valatavičius, A. (2020). Extending model-driven development process with causal modeling approach. In: Dzemyda, G., Bernatavičienė, J., Kacprzyk, J. (Eds.), *Data Science: New Issues, Challenges and Applications*, *Studies in Computational Intelligence*, Vol. 869. Springer Nature, Switzerland, pp. 279–296. https://doi.org/10.1007/978-3-030-39250-5_7.

Gudas, S., Noreika, K. (2022). Causal interactions in Agile application development. *Mathematics*, 10(9), 1497. https://doi.org/10.3390/math10091497.

Gudas, S., Tekutov, J., Butleris, R., Denisovas, V. (2019). Modelling subject domain causality for learning content renewal. *Informatica*, 30(3), 455–480. https://doi.org/10.15388/Informatica.2019.214.

Guta, G., Schreiner, W., Draheim, D. (2009). A lightweight MDSD process applied in small projects. In: *35th Euromicro Conference on Software Engineering and Advanced Applications*. IEEE, USA, pp. 255–258. https://doi.org/10.1109/SEAA.2009.63.

Kirby, J. Jr. (2006). Model-driven Agile development of reactive multi-agent systems. In: *COMPSAC'06*. IEEE, USA, pp. 297–302.

Kleppe, A., Warmer, J., Bast, W. (2003). *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison Wesley, USA.

Knapp, J. (2016). *Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days*. 1st ed., Simon & Schuster, USA.

KPMG, AIPM, IPMA (2019). The Future of Project Management: Global Outlook 2019. https://www.ipma.world/assets/PM-Survey-FullReport-2019-FINAL.pdf. Last accessed 2022/03/02.

Kulkarni, V., Reddy, S. (2004). Model-driven development of enterprise applications. In: Nunes, N.J., Selic, B., da Silva, A.R., Alvarez, A.T. (Eds.), *UML Modeling Languages and Applications, UML 2004, Satellite Activities*, *LNCS*, Vol. 3297. Springer-Verlag, Berlin, Heidelberg, pp. 118–128.

Kulkarni, V., Barat, S., Ramteerthkar, U. (2011). Early experience with Agile methodology in a model-driven approach. In: Whittle, J., Clark, T., Kühne, T. (Eds.), *MODELS 2011: Model Driven Engineering Languages and Systems*, *LNCS*, Vol. 6981. Springer-Verlag GmbH, Berlin, Heidelberg, New Zealand, pp. 578–590. https://doi.org/10.1007/978-3-642-24485-8.

Lano, K., Alfraihi, H., Yassipour Tehrani, S., Haughton, H. (2015). Improving the application of agile modelbased development: experiences from case studies. In: *The Tenth International Conference on Software Engineering Advances*. IARIA, Spain, pp. 213–219.

Li, J., Horiguchi, Y., Sawaragi, T. (2022). Counterfactual inference to predict causal knowledge graph for relational transfer learning by assimilating expert knowledge – relational feature transfer learning algorithm. *Advanced Engineering Informatics*, 51, 101516. https://doi.org/10.1016/j.aei.2021.101516.

Matinnejad, R. (2011). Agile model driven development: an intelligent compromise. In: *SERA*. IEEE, USA, pp. 197–202.

Miller, L. (2005). Case study of customer input for a successful product. In: *ADC'05*. IEEE, USA, pp. 225–234. https://doi.org/10.1109/ADC.2005.16.

Nakicenovic, M.B. (2012). An agile driven architecture modernization to a model-driven development solution. *International Journal on Advances in Software*, 5(3–4), 308–322.

Noreika, K. (2021). Improving enterprise application software development management with MODAF. In: Forbrig, P., Hinkelmann, K., Kirikova, M., Lantow, B., Møller, C., Morichetta, A., Plebani, P., Re, B., Sandkuhl, K., Seigerroth, U. (Eds.), *Joint Proceedings of the BIR 2021 Workshops and Doctoral Consortium Co-located with 20th International Conference on Perspectives in Business Informatics Research (BIR 2021)*, Vienna, Austria, 2021, pp. 141–152. https://ceur-ws.org/Vol-2991/paper12.pdf.

Noreika, K., Gudas, S. (2021). Modelling the alignment between Agile application development and business strategies. In: *Joint Proceedings of the BIR 2021 Workshops and Doctoral Consortium Co-located with 20th International Conference on Perspectives in Business Informatics Research (BIR 2021)*, Vienna, Austria, September 22–24, 2021, Vienna, pp. 59–73.

The OMG (2022a). MDA – The architecture of choice for a changing world. https://www.omg.org/mda/. Last accessed 2022/11/06.

The OMG (2022b). Business modeling category – specifications associated. https://www.omg.org/spec/category/business-modeling/About-business-modeling/. Last accessed 2022/11/06.

Porter, M.E. (1985). *Competitive Advantage*. 1st ed., The Free Press, New York, USA.

Prior, D. (2022). Agile Planning with Ties with Tom Churchwell. https://www.leadingagile.com/podcast/agile-planning-ties-tom-churchwell/. Last accessed 2022/08/03.

Rivero, J.M., Luna, E.R., Grigera, J., Rossi, G. (2013). Improving user involvement through a model-driven requirements approach. In: *2013 3rd International Workshop on Model-Driven Requirements Engineering (MoDRE)*, Rio de Janeiro, Brazil, 2013, pp. 20–29. https://doi.org/10.1109/MoDRE.2013.6597260.

Rivero, J.M., Grigera, J., Rossi, G., Luna, E.R., Montero, F., Gaedke, M. (2014). Mockup-driven development: providing agile support for model-driven web engineering. *Information and Software Technology*, 56(6), 670–687.

Robles Luna, E., Grigera, J., Rossi, G. (2009). Bridging test and model-driven approaches in web engineering. In: Gaedke, M., Grossniklaus, M., Díaz, O. (Eds.), *ICWE 2009: Web Engineering*, *LNCS*, Vol. 5648. Springer, Berlin, Heidelberg, Spain, pp. 136–150.

Zhang, Y., Patel, S. (2011). Agile model-driven development in practice. *IEEE Software*, 28(2), 84–91.

K. Noreika obtained master's degree in 2015 from Vilnius University. He is a PhD student at Vilnius University Institute of Data Science and Digital Technologies. His research interests include Agile software development management, business and IT alignment methods. K. Noreika has published an article in the cited journal (indexed by the Web of Science database) and also papers in peer-reviewed conference proceedings (indexed by Scopus). He is a member of the Lithuanian Computer Society. Karolis has over 14 years of experience in Agile software development management working as project manager, business analyst, and Agile consultant in national and international companies.

S. Gudas is a head of the Cyber-Social Systems Engineering Department, professor at the Faculty of Mathematics and Informatics of Vilnius University. He received a doctoral degree in technical sciences (PhD) in 1982. In 2005 he completed the habilitation procedure in informatics engineering at Kaunas University of Technology. He was conferred the title of professor at the Kaunas University of Technology (2005), and at Vilnius University (2005). He is the author of the monograph *Foundations of the Information Systems Engineering Theory*, 2 book chapters, 5 textbooks, and more than 180 scientific papers. The field of research interests focuses on enterprise software engineering, causal modelling approach, knowledge-based CASE methods. He is a supervisor of eight PhD theses. Member of Lithuanian Computer Society. Scientific internship: Warsaw Technical University, Poland, 2018; Riga Technical University, Latvia, 2016; Tartu University, Estonia, 2015; Brandenburg University of Technology (Germany, Cottbus-Senftenberg), 2013, London City University, 1996; Stockholm University, 1995.