Utilization of Asset Adminstration Shells for e-Commerce Applications

Markus Rentschler
ARENA2036 e.V., Germany
markus.rentschler@arena2036.de

Alwin Hoffmann
XITASO GmbH, Germany
alwin.hoffmann@xitaso.com

Moritz Hofer
XITASO GmbH, Germany
moritz.hofer@xitaso.com

***Abstract*—There are a multitude of electronic marketplaces that facilitate buyers and sellers in their transactions over the Internet. For Business-to-Business (B2B) transactions, related EDI standards are well established to handle system interoperability in the e-Commerce process. But in Business-to-Customer (B2C) transactions, from the end-user perspective, there is often a lock-in experience to the dedicated software application provided by the respective electronic marketplaces, without the possibility to use a single front-end application to handle business in different electronic shop systems. Standardized interfaces for this purpose are not sufficiently defined or established. In this work, the asset administration shell technology is evaluated in providing such a standardized interface, to achieve system interoperability of marketplaces towards end-customer applications.**

***Index Terms*—AAS, asset administration shell, BMEcat, eCommerce, e-Commerce, e-Shop, headless commerce, marketplace, online shop, pricing**

I. Introduction

Electronic commerce (e-Commerce) has becoming widespread in B2C and B2B shopping scenarios and a wide range of information technology systems support e-Commerce processes. Reference models have been designed for such systems, covering business processes and related enterprise architecture [1]. The key use cases for an e-Commerce system are as follows:

1) **Catalog Browsing and Search**: Customers browse product catalogs or search for specific items using filters (e.g., category, price, brand) or keywords.
2) **Shopping Cart Operations**: Customers add, remove, or modify items in their shopping cart.
3) **Checkout Process**: Customers finalize their purchases, including entering shipping and payment details.
4) **Payment Processing**: Customers handle secure transactions through various payment methods (e.g., credit cards, digital wallets).
5) **Order Management**: Customers process orders, track order status, and handle post-purchase actions such as cancelations or returns.
6) **Customer Support Services**: Customers are provided assistance through channels such as live chat, email, or ticketing systems for issues such as refunds, product inquiries, or order disputes.
7) **Customer Account Management**: Customers create and manage accounts, view order history, save preferences, and manage payment or shipping information.
8) **Product Information Management**: Administrators manage and provide detailed product data, such as descriptions, specifications, prices, and images.
9) **Analytics and Reporting**: Administrators have insights on sales performance, customer behavior, and inventory trends for business decision-making.
10) **Marketing and Promotions**: Administrators manage promotional campaigns, also with discounts and personalized offers based on customer behavior or purchase history (dynamic pricing).

The interoperability between different e-Commerce systems is challenging, and a range of EDI-Standards[1] emerged over the years to provide suitable interfaces.

EDI protocol standards such as AS2, AS4, SFTP, VANs, and ebXML meet the need for secure and reliable transport mechanisms for EDI data standards such as EDIFACT, UBL cXML, openTRANS and BMEcat, enabling the automation of orders, invoices, and logistics in e-Commerce, particularly for B2B transactions [2]. Furthermore, standards such as GS1, UNSPSC, ECLASS or ETIM serve the purpose of structuring and semantically referencing product, order, and customer data [3].

The Asset Administration Shell (AAS) technology was designed with the potential to integrate other standards [4] and offers a file- [5] and API-based interface [6]. Thus, it could potentially serve as a unified framework for seamless EDI integration with third-party services and support architectures such as microservices or *headless commerce*[2], an approach in e-Commerce which decouples front-end and back-end systems. In this paper, an analysis is conducted regarding the suitability of AAS technology for e-Commerce systems, with closer regard on the B2C part of the infrastructure, where customers interact with the respective marketplaces. The emerging new submodel template *IDTA 02050-1-0 Purchase Order* [7] is analyzed for suitability and necessary enhancements proposed.

II. Related Work

Business information such as product catalogues, purchase orders, invoices and shipping notices were traditionally communicated on paper. The concept of electronically communicating such business information has existed at least since the

[1] https://en.wikipedia.org/wiki/Electronic_data_interchange
[2] https://www.shopify.com/enterprise/blog/headless-commerce

early 1970s and many technical EDI standards have since been developed, some addressing the needs of specific industries or regions. Some possibly relevant EDI standards for the context of this work are mentioned below and Table I summarizes in an overwiew.

A. GS1

GS1 EDI standards (EANCOM, GS1 XML, GS1 UN/XML enable automated, standardized, and accurate exchange of transactional data across the supply chain. GS1 EDI supports dynamic processes such as ordering, shipping, and invoicing.[3]

B. EDIFACT

EDIFACT (Electronic Data Interchange for Administration, Commerce and Transport) according to ISO 9735 is a global standard, defining message formats for orders, invoices and shipping notices, focusing on the content and syntax of the data itself.[4]. EDIFACT is commonly used in legacy large-scale e-Commerce systems integrated with ERP platforms, and VDA 4901 [8] explicitly recommends the use of EDIFACT messages for supply chain processes in the automotive industry.

EANCOM[5] is a subset of EDIFACT, specifically tailored for the retail and supply chain sectors. Developed by GS1, it simplifies and adapts EDIFACT's complex syntax to meet the needs for efficient B2B communication in areas such as ordering, billing, and inventory management.

C. UBL

UBL (Universal Business Language) is an XML-based standard for business documents such as invoices, purchase orders, and catalogs, maintained as part of the OASIS ebXML Documentation[6]. UBL is widely adopted in Europe for public sector e-Commerce and the core standard for PEPPOL (Pan-European Public Procurement Online), a specification framework for cross-border e-Invoicing.[7]

D. cXML

cXML (Commerce XML)[8] is a protocol for real-time B2B e-Commerce transactions, focusing on purchase orders, invoices, and catalogs. cXML is used by proprietary platforms such as SAP Ariba[9] for supplier integration, supporting dynamic pricing and inventory updates.

E. BMEcat

BMEcat[10] was developed in 1999 by the German Federal Association for Materials Management, Purchasing and Logistics (BME) as XML-based exchange format to standardize the transfer of product catalog data between suppliers and buyers in the B2B sector.

[3] https://www.gs1.org/standards/edi
[4] https://unece.org/trade/uncefact/unedifact
[5] https://www.gs1-germany.de/standards/datenaustausch/eancom/
[6] https://www.oasis-open.org/standards/#ebxml
[7] https://peppol.eu/
[8] https://cxml.org/
[9] https://en.wikipedia.org/wiki/SAP_Ariba
[10] https://www.bme.de/services/bmecat/

Key features include support for multilingual data, classification systems, and custom fields via `USER_DEFINED_EXTENSIONS`. Classification systems like ECLASS or ETIM are often used to structure product data. BMEcat supports transactions like `T_NEW_CATALOG`, `T_UPDATE_PRODUCTS`, and `T_UPDATE_PRICES`, to be processed in the receiving system.

BMEcat allows to manage pricing information as outlined below:

Structured Pricing Elements: `ARTICLE_PRICE` and `ARTICLE_ORDER_DETAILS` includes fields like `ORDER_UNIT`, `CONTENT_UNIT` and `NO_CU_PER_OU`.

Currency and Tax Information: `PRICE_CURRENCY` allows association of prices with a specific currency. Tax information, such as VAT rates, can be included using the element `TAX` to ensure compliance with regional requirements.

Multiple Price Types are supported by `PRICE_TYPE`, such as list prices, customer-specific prices, discounted prices, or promotional prices with attributes like `PRICE_TYPE` (e.g., net_list, net_customer, gross_list).

Dynamic Pricing Components: Dynamic pricing models, like tiered pricing based on quantity or customer segments, allow suppliers to tailor prices for personalized B2B contracts. For example, the `PRICE_QUANTITY` element defines the quantity to which a price applies, and `QUANTITY_MIN` and `QUANTITY_INTERVAL` specify minimum order quantities and increments.

Version Control for Price Updates: The transaction `T_UPDATE_PRICES` allows to update pricing information without modifying other product data, enabling suppliers to communicate price changes while maintaining a version history to the price updates.

Listing 1: Example XML for BMEcat

```
<ARTICLE>
    <ARTICLE_ID>12345</ARTICLE_ID>
    <ARTICLE_PRICE price_type="net_list">
        <PRICE_AMOUNT>10.50</PRICE_AMOUNT>
        <PRICE_CURRENCY>EUR</PRICE_CURRENCY>
        <TAX>0.19</TAX>
        <PRICE_QUANTITY>100</PRICE_QUANTITY>
    </ARTICLE_PRICE>
    <ARTICLE_ORDER_DETAILS>
        <ORDER_UNIT>PK</ORDER_UNIT>
        <CONTENT_UNIT>C62</CONTENT_UNIT>
        <NO_CU_PER_OU>500</NO_CU_PER_OU>
    </ARTICLE_ORDER_DETAILS>
</ARTICLE>
```

In this BMEcat example, the product (ID 12345) has a net list price of 10.50€ per 100 units, with a 19% tax rate, and is ordered in packs containing 500 units.

TABLE I: Overview on EDI Data Standards

Aspect	GS1	EDIFACT	UBL	cXML	BMEcat	openTRANS
Format	XML, EANCOM	Text-based	XML	XML	XML	XML
Transport	AS2, FTP, HTTP, VAN	AS2, FTP, SFTP, VAN	AS2, AS4, HTTP	HTTP, HTTPS	HTTP, FTP, SFTP	FTP, HTTP
Primary Use	Product ID, supply chain	Business document exchange	e-Invoicing, procurement	e-Procurement, catalogs	Product catalog exchange	Transactional documents (e.g., orders, invoices)
Key Strength	Global product identification	International trade compatibility	EU-compliant e-Invoicing	Real-time procurement integration	Detailed product data	Digital payment support
Examples	Barcode on products, EDI orders	Invoice for cross-border trade	Government e-Invoice	Punch-out catalog access	Supplier product catalog	Procurement automation
Pricing Semantics	EANCOM: PRI for unit prices, ALC for discounts; GS1 `<price>` for base/promotional prices	PRI for unit prices, ALC for discounts/charges, TAX for taxes	`<Price>`, `<PriceAmount>`, `<AllowanceCharge>` for prices/discounts	`<UnitPrice>`, `<Money>` for catalog/order prices	`<ARTICLE\_PRICE>`, `<PRICE\_TYPE>` for catalog pricing, supports tiers	pricing attributes like BMEcat (e.g., price_type, currency)
Adoption	Global, esp. retail and logistics (e.g., Walmart, Amazon)	Global, esp. in trade and logistics (UN-supported)	Strong in EU public sector, e-commerce	Common in procurement platforms (e.g., SAP Ariba)	Strong in Europe, esp. Germany (manufacturing, retail)	Germany (industrial)

F. openTRANS

openTRANS[11] was developed by the German Federal Association for Materials Management, Purchasing, and Logistics (BME) as a XML-based standard for transactional business documents, such as purchase orders, order confirmations, invoices and delivery notes during the procurement lifecycle. It facilitates automated procurement processes between buyers and suppliers and complements the BMEcat standard, with which it shares a common XML-based framework: BMEcat handles product catalog data, while openTRANS manages transactional data. Together, they form an integrated solution for end-to-end procurement automation and relate as follows:

BMEcat provides the foundation by supplying standardized product catalog data, including:

- Product IDs, names, descriptions, and technical attributes.
- Pricing information (e.g., list prices, discounts).
- Classification systems (e.g., ECLASS, UNSPSC).
- Media files (e.g., images, datasheets).

openTRANS builds on this by enabling the transactional processes that use the catalog data, such as:

- Placing purchase orders (ORDER) based on products selected from a BMEcat catalog.
- Sending order confirmations (ORDERRESPONSE) or invoices (INVOICE) referencing catalog items.

Example: A buyer uses a BMEcat catalog to select products in their ERP system, then generates an openTRANS purchase order to send to the supplier.

Listing 2: Example XML for openTRANS

```
<ORDER_ITEM>
  <LINE_ITEM_ID>001</LINE_ITEM_ID>
  <PRODUCT_ID>
    <SUPPLIER_PID>12345</SUPPLIER_PID>
  </PRODUCT_ID>
  <QUANTITY>100</QUANTITY>
  <PRICE type="net_customer" currency="EUR">
      10.50</PRICE>
  <PRICE_QUANTITY>1</PRICE_QUANTITY>
  <PRICE_AMOUNT>1050.00</PRICE_AMOUNT>
  <DISCOUNT>
    <DISCOUNT_VALUE type="percentage">5</
        DISCOUNT_VALUE>
    <DISCOUNT_REASON>Volume Discount</
        DISCOUNT_REASON>
  </DISCOUNT>
  <TAX>
    <TAX_RATE>19</TAX_RATE>
    <TAX_AMOUNT>199.50</TAX_AMOUNT>
  </TAX>
  <TOTAL_ITEM_AMOUNT>1249.50</
      TOTAL_ITEM_AMOUNT>
</ORDER_ITEM>
```

This openTRANS example shows a simplified ORDER_ITEM with pricing semantics (unit price of €10.50, a 5% discount, 19% VAT, and a total item amount).

III. Analysis

A. IDTA Purchase Order

The emerging standard IDTA 02050-1-0 Purchase Order Submodel Template [7] was designed as a framework for standardizing data for certain procurement transactions (purchase requests, quotations, orders) in smart manufacturing. Based on AAS technology and focusing on interoperability and semantic clarity, it facilitates B2B transactions within the AAS ecosystem. It is based on previous work in the context of the INTEROPERA project[12]. Analysis showed that IDTA 02050-1-0 explicitly references UN/EDIFACT, cXML/SAP Ariba, and openTRANS in concept descriptions, bridging traditional EDI with Industry 4.0. Obviously concepts from these standards as well as proprietary ecosystems are adopted. In fact, many of the definitions align with the SAP Ariba cXML interface. Semantic interoperability using ECLASS and IEC CDD is emphasized. Table II summarizes the semantic alignment for key purchase order concepts across the standards. Although terminology is slightly different, it is obvious

[11] https://de.wikipedia.org/wiki/OpenTRANS

[12] https://interopera.de/teilmodellprojekte/

TABLE II: Semantic Alignment Table

Semantic Concept	GS1	UN/EDIFACT	UBL	cXML	BMECat	openTRANS	IDTA 02050-1-0
Document Type	order, transactionType (e.g., ORIGINAL)	BGM (e.g., 220 for order)	Order, OrderTypeCode	OrderRequest, type (e.g., new, update‘)	T_NEW_CATALOG (no orders)	ORDER, QUOTATION, ORDER_TYPE	DocumentType (e.g., PurchaseRequest, Quotation, Order)
Product ID	gtin, buyerProductCode	PIA (e.g., SA for supplier’s ID)	Item, SellersItemIdentification	ItemID, SupplierPartID	ARTICLE_ID, SUPPLIER_AID	PRODUCT_ID, SUPPLIER_PID	Item:ItemID
Pricing	netPrice, grossPrice, totalLineAmount	PRI, MOA, ALC (allowances)	Price, AllowanceCharge, LineExtensionAmount	Money, UnitPrice, Total, DiscountAmount	ARTICLE_PRICE, PRICE_SCALE	PRICE, DISCOUNT, TOTAL_AMOUNT	TotalAmount, CustomsAmount
Quantities	orderedQuantity, measurementUnitCode	QTY (e.g., 21 for ordered)	Quantity, QuantityUnitCode	ItemIn, quantity, UnitOfMeasure	ORDER_UNIT, CONTENT_UNIT	QUANTITY, UNIT	Item:OrderQuantity
Parties	gln (e.g., buyer, seller)	NAD (e.g., BY for buyer)	Party, PartyIdentification	Contact (e.g., buyer, supplier)	BUYER, SUPPLIER	PARTIES, PARTY	InvolvedParties (e.g., BuyerPurchaseOrderNumber)
Taxes	taxAmount, taxRate	TAX, MOA	TaxTotal, TaxSubtotal	Tax, TaxDetail	TAX (catalog-level)	TAX, TAX_RATE	TaxDetails
Delivery Terms	requestedDeliveryDate, shipTo	DTM, LOC, TOD (Incoterms)	Delivery, RequestedDeliveryPeriod	ShipTo, RequestedDeliveryDate	DELIVERY_TIME	DELIVERY_DATE, SHIP_TO	DeliveryInformation (e.g., ValidityStartDate)
Payment Terms	paymentTerms (e.g., NET_DAYS)	PAT, PAI (e.g., bank transfer)	PaymentTerms, PaymentMeansCode	PaymentTerm, Payment	None	PAYMENT_TERMS, DISCOUNT	PaymentDetails (e.g., CardType)
Tracking and Logistics	sscc (barcodes, RFID)	RFF (tracking), PCI (package)	Shipment, TrackingID	Extrinsic (custom)	LOGISTIC_ DETAILS	TRACKING_CODE (e.g., NFC, QR)	BuyerTrackingCodeList (e.g., NFC, QR-code)
Order Status	orderStatus (e.g., ACCEPTED)	STS (e.g., 1 for accepted)	OrderResponse, StatusCode	OrderStatus, status	None	ORDER_STATUS	Status (e.g., Status_isOpen)
Vocabularies	GS1-specific, ECLASS	UN/EDIFACT-specific, UN/CL	UBL-specific, UN/CL, PEPPOL	cXML-specific, some UN/CL	BMECat-specific, ECLASS	openTRANS, BMECat-aligned	cXML, openTRANS, UN/EDIFACT, ECLASS

that the property definitions are easily mapped, although may not always fit exactly.

B. Submodel Review

The new *IDTA 02050-1-0 Purchase Order* [7] supports the procurement process in the aspects of *PurchaseRequest*, *Quotation* and *PurchaseOrder*. In addition to supporting these procurement transactions, its data model basically allows one to represent a list of product items together with pricing information as well as payment options. These elements are also the basics of a product catalog and are represented within the SMC *PurchaseOrderInformation*, SML *ItemList*, SMC *ProductPrice*, SMC *InvolvedParties/PartySeller* and the SML *ModeOfPaymentList* (see Figure 4).

In the current draft of *IDTA 02050-1-0 Purchase Order*, the following data definitions appear to be limiting, redundant, or inconsistent and therefore are proposed to be corrected.

- *BuyerReferenceID*, *SellerReferenceID*, *AgentReferenceID* were declared for every entity of *Party*, which seems to be erroneous and should instead be corrected with a single property *ReferenceID* instead.
- The content of SMC *InvolvedParties* should be as semantically aligned to the SM *IDTA 02068 Company Data* [9] and the SM *IDTA 02002 Contact Informations* [10].
- SML *AttachedMediaList* should allow semantic classification of documents similar to SM *IDTA 02004 HandoverDocumentation* [11]. For this purpose, the properties *File* and *MimeType* could be replaced by a single property *MediaFile* of type *[File]*, whereas property *MimePurpose* could be remodeled to a SMC *DocumentClassification* as in SM *IDTA 02004 HandoverDocumentation*, and ”IEC61355-1:2008”[13] be recommended as classification system.
- Cardinality should rather be *One* for *PurchaseOrderInformation/Language* and *ZeroToOne* for *BuyerPurchaseOrderNumber*, *PurchaseOrderType*, *InvolvedParties/PartyBuyer*, *OrderQuantity*, *ScopeOfDelivery*, *PriceUnitValue*, *ProductShortDescription* and *ProductDescription* (since the latter are potentially redundant to content of *AASItemReference*).
- The terminology of some property could be improved by renaming *IncoTerms/Location1* to *SupplierLocation*, *IncoTerms/Location2* to *BuyerLocation* and *IncoTerms/Location3* to *ShippingLocation*.
- The applicability of *PriceQuantity*, *PriceUnitValue* is unclear, and their definition texts should be improved.

[13] https://en.wikipedia.org/wiki/IEC_61355

C. System Context

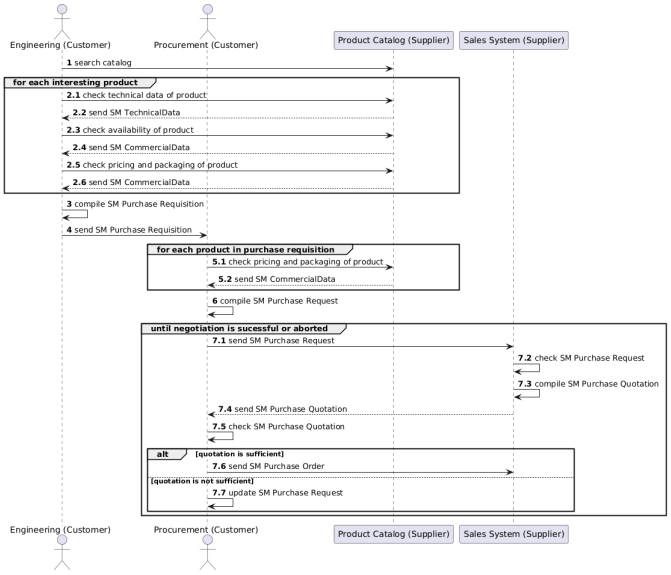


Fig. 1: Purchase activities

From our point of view, the scope of the submodel *IDTA 02050-1-0 Purchase Order* misses a fundamental part of an e-Commerce workflow. In Fig. 1, we show an exemplary workflow for ordering in a B2B setting. Someone in the engineering or design department is looking for appropriate out-of-shelf components to create a product in an AAS-based product catalog (see 1, 2.1, and 2.2 in Fig. 1). Therefore, the person checks the technical properties (represented by the submodel *Generic Frame for Technical Data for Industrial Equipment in Manufacturing* [12]) and also the documentation (represented by the submodel *Handover Documentation* [11]). Furthermore, the person needs to check availability, the packaging options and also the pricing in order to determine if it fits into the overall product idea (see 2.3-2.6 in Fig. 1).

If everything is satisfactory, the next step is to hand it over to the procurement department as specified by the submodel *Purchase Order* for out-of-shelf products. First, engineering creates a Purchase Request for the procurement (see 3-4 in Fig. 1). In the procurement department, the pricing is checked again (see 5.1-5.2 in Fig. 1) and finally the negotiation is started (see 6 and 7 in Fig. 1).

For ordering out-of-shelf products, the submodel draft *Purchase Order* is well suited as the workflow of 6 and 7 in Fig. 1 shows. However, an essential part is still missing: commercial data on the out-of-shelf products available in a product catalog. This includes pricing and packaging information as well as availability. Hence, we propose to improve the submodel *IDTA 02050-1-0 Purchase Order* in Sect. IV to include the missing e-Commerce information.

IV. Solution

A. Submodel Enhancements

The forthcoming Submodel *IDTA-02050-1-0 Purchase Order* [7] facilitates quotation and order processing, but lacks capabilities for handling pre-sales information within product catalogs and e-Shop transactions. To address this deficiency, we propose an enhanced version of the Submodel. In pursuit of broader applicability, we suggest renaming SM *PurchaseOrder* to *SM CommercialData* or a similar designation, and SMC *PurchaseOrderInformation* to SMC *DocumentInformation*.

The current limitation of ***DocumentType*** to transactions ***PurchaseRequest***, ***Quotation***, and ***Order*** should be expanded by including additional transaction types pertinent to product catalogs and e-Shop systems:

- ***ProductPriceInfo***: Provide commercial information for one or more items in the product catalog for various quantities. Also applicable to communicate price changes.
- ***AddToCart***: Add a catalog item to an order list in a remote e-Shop system.
- ***UpdateCart***: Remove or change a catalog item on an order list in a remote e-Shop system.
- ***ShowCart***: Displays the current contents of an order list from a remote e-Shop system.
- ***ShowOrderStatus***: Shows the current status of an order from a remote e-Shop system.

Figure 4 illustrates the proposed modifications to the *PurchaseOrder* Submodel Template, highlighted in blue. Our modifications are categorized into “Additional Payment Options”, “Usage of Already Standardized Patterns”, “Additional Item Attributes for e-Commerce”, and “Additional Shipment Information”, as detailed in the following sections.

1) Additional Payment Options: For e-Commerce applications, it is essential to provide information on purchasing methods from the seller. Although the current draft of the *PurchaseOrder* submodel template supports a list of *PreferredModeOfPayment* objects, it requires enhancement to inform potential buyers about the available payment types per country and the possibility of financing through a payment provider. We have extended *PreferredModeOfPayment* with relevant properties, as shown in Figure 4.

2) Usage of already Standardized Patterns: The *PurchaseOrder* submodel template requires information that was standardized in other submodels. Specifically, for *InvolvedParties*, we strongly recommend reusing existing patterns, such as drop-ins from the Submodel Templates *HandoverDocumentation* [11], *ContactInformations* [10] and *CompanyData* [9]. This is indicated in Figure 4 with the newly added *MediaClassification*, *PartyContactInformation* and *PartyCompanyIdentification* . In anticipation of the requirements of the European ESPR Regulation, the property *UniqueFacilityIdentifier* as defined in *IDTA-02006-3-0 Digital Nameplate* [13] could also be adopted and added to the SMC *Party*.

3) Additional Item Attributes for e-Commerce: To utilize the *Item* object for product price information in online product catalogs and e-Shops, we propose the following enhancements. Users need to know the permissible order quantities, *Minimum/MaximumOrderQuantity*, the required order interval, e.g., multiples of 4 - *QuantityInterval*, and the product’s origin, *CountryOfOrigin*. Information on availabil-

ity(*CurrentlyAvailable*), and delivery time(*DeliveryTime*), aids in informed decision-making.

Catalog providers also require information on the availability of additional documentation, such as chain of custody, digital product passport, or engineering documents. Keywords facilitate faster and more precise item searches in product catalogs.

To ensure accurate product price information, we propose changing it to a *SubmodelElementList* with multiple *ProductPrice* objects. This *ProductPrice* can represent the `ProductPrice` from the BMECat standard, see section II-E, enabling customer-specific pricing per order quantities or special prices upon request.

4) Additional Shipment Information: Depending on the quantity of ordered products, the shipment can be in different packaging units, e.g., one bottle, one six-pack, one pallet. To model these relationships, a new element *PackagingUnits* was introduced, defining a new packaging unit with possible shipments based on the specified order quantity.

These shipments can be modeled in the *DeliveryDate* element, which we expanded to include shipping costs, terms, and details.

It should be noted that the AAS that contains the more dynamic commercial data according to the proposed submodel template is probably hosted on a dedicated AAS server closely related to a company's sales system, which is an ERP or warehouse system. In such a multiple AAS server scenario, the cross-identification between the different AAS for the static technical data and more dynamic commercial data for the same product must be organized with suitable AAS-ID definitions.

B. Process View

The enhanced submodel shown in Figure 4 contains sensitive data intended for specific customers, like `ProductPrice` with `PriceType == net_customer`. It is crucial to ensure that the right party receives the correct data. Therefore, a secure method for retrieving and sending this e-Commerce Information submodel is necessary. We suggest storing this e-Commerce submodel in a separate repository with enhanced security, as specified in the AAS Specification *Part 4: Security (IDTA-01004)* [14].

Figure 2 shows how a product catalog interacts with a user. All Asset Administration Shells (AAS) and Submodels are kept in publicly available repositories, allowing users to integrate the data directly into their systems. However, the e-Commerce submodel for a product catalog is usually stored in a separate repository. A Submodel Descriptor, found via the registry, points to this e-Commerce AAS-Repository. Security checks verify if users are authorized to access the data, prompting unauthorized users to log in, or providing authorized users with the data they are allowed to see. Possible scenarios include:

1) All users can see all pricing information: The e-Commerce Submodel Repository returns the complete e-Commerce Submodel.
2) All users can see prices, but specific users can see reduced customer prices: The repository returns the e-Commerce Submodel and adds customer-specific prices for authorized users.
3) Only logged-in users can see any prices: The repository returns an unauthorized exception or proceeds as in Scenario 2.
4) No one can see prices; everyone needs to contact the supplier: The repository returns the e-Commerce Submodel with `PriceVisibility == on_request`.

This customized information delivery can also be applied to other variable data, such as `ModeOfPayment`, `MaximumQuantity`, or `ShippingDetails`.

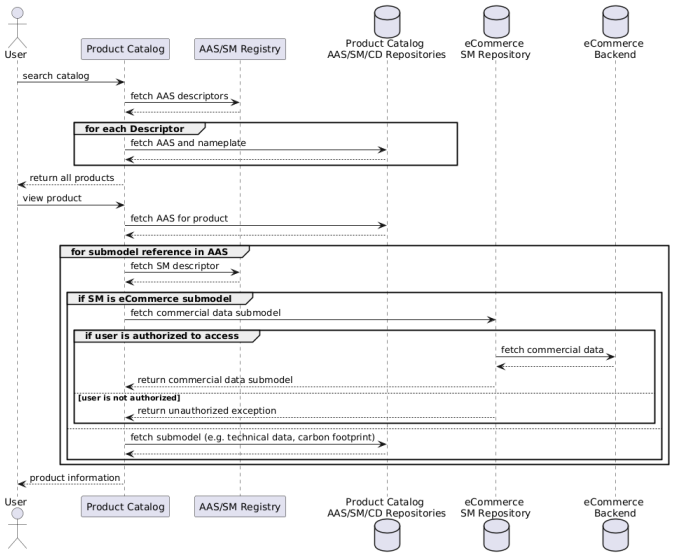


Fig. 2: Scenario for product catalog with price information

C. Proof of Concept

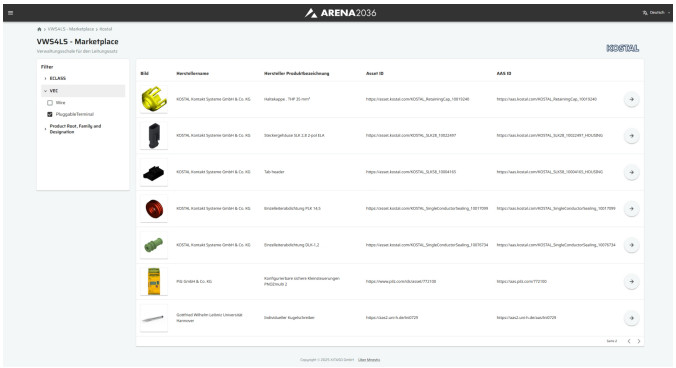
As part of the VWS4LS[14] project, a product catalog demonstrator[15] based on AAS technology was developed as a proof of concept using Eclipse Mnestix[16].

In Fig. 3a, the proof-of-concept for a product catalog can be seen. This catalog displays all product catalog AAS from an AAS repository, which are of `assetKind == type` and come with a technical data submodel, making it filterable by attributes of the AAS, especially on product classification and technical data properties. In Fig. 3b, it is indicated on how the pricing information retrieved from the associated Commercial Data submodel (see section 4) could be displayed, for the user to understand different pricing modalities for the product.

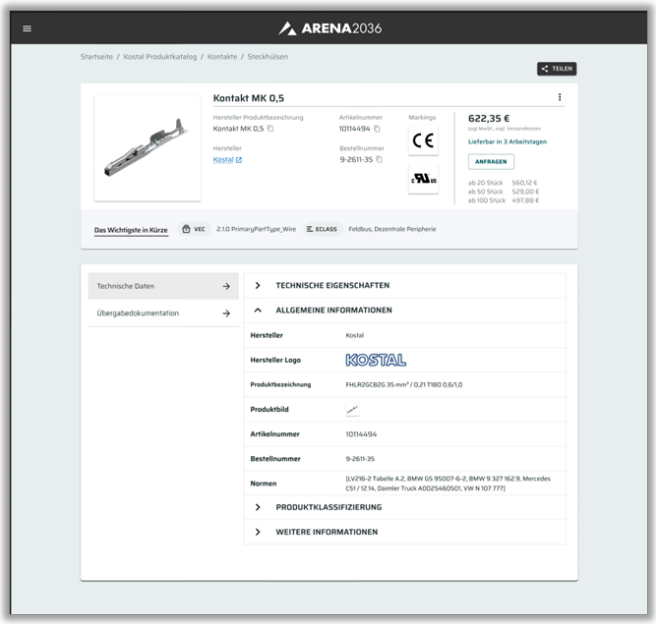
V. Conclusion

An overview of relevant EDI standards was collected and the emerging *IDTA 02050-1-0 Purchase Order* submodel template was analyzed for suitability to represent the necessary

[14]https://github.com/VWS4LS
[15]https://marketplace.arena2036.app
[16]https://github.com/eclipse-mnestix/mnestix-browser



(a) Mnestix product catalog view



(b) Mnestix product view with commercial data

Fig. 3: Proof of Concept in Eclipse Mnestix Browser

commercial data for online product catalogs and e-Shop-Backends. In the review process of the draft document, some general shortcomings were identified and improvement proposals made. The analysis showed that the semantic alignment of *IDTA 02050-1-0 Purchase Order* with *cXML*, *BMEcat*, and *openTRANS* is strong for core purchase order concepts (e.g., pricing, quantities, parties), and only a few information elements are missing to achieve the desired applicability for product catalogs and e-Shop transactions. To close this gap, some enhancements and changes to the submodel were defined to achieve this enhanced applicability, and based on this, an adopted submodel template "*CommercialData*" proposed.

A prototypical implementation of a marketplace user interface based on AAS technology and the open source Eclipse-based AAS-Viewer *Mnestix* was developed to demonstrate the feasibility of the approach.

VI. Acknowledgement

The authors appreciate the support from the *German Federal Ministry for Economic Affairs and Climate Action* (BMWK) through the VWS4LS project (Grant No. 13IK005A). In addition, thanks are given to the many active participants in the project and the working groups of the Industry 4.0 Platform and related initiatives.

References

[1] F. Aulkemeier, M. Schramm, M.-E. Iacob, and J. Van Hillegersberg, "A service-oriented e-commerce reference architecture," *Journal of theoretical and applied electronic commerce research*, vol. 11, no. 1, pp. 26–45, 2016. [Online]. Available: https://doi.org/10.4067/S0718-18762016000100003

[2] G. Hohpe and B. Woolf, *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Boston, MA: Addison-Wesley Professional, 2003.

[3] G. Samtani and P. Sadhwani, *Modern B2B Integration: Strategies for Connecting Applications, Data, and Devices in the Modern Enterprise*. Boise, ID: MC Press, 2002.

[4] R. Drath, M. Rentschler, and M. Hoffmeister, "The AutomationML Component Description in the context of the Asset Administration Shell," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 1278–1281.

[5] "Asset Administration Shell Specification Part 5: Package File Format (AASX)," Industrial Digital Twin Association, Germany, Tech. Rep., 2024. [Online]. Available: https://industrialdigitaltwin.io/aas-specifications/IDTA-01005/v3.1/index.html

[6] "Asset Administration Shell Specification Part 2: Application Programming Interfaces," Industrial Digital Twin Association, Germany, Tech. Rep., 2024. [Online]. Available: https://industrialdigitaltwin.io/aas-specifications/IDTA-01004/v3.0/index.html

[7] "IDTA 02050-1-0 Submodel Template - Purchase Order," Industrial Digital Twin Association, Germany, Tech. Rep., 2025, under review.

[8] "Grundbegriffe zum Datenaustausch in der Lieferkette V1.0," Verband der Automobilindustrie (VDA), Berlin, Germany, Tech. Rep., December 2019. [Online]. Available: https://www.vda.de/de/aktuelles/publikationen/publication/vda-4901---grundbegriffe-zum-datenaustausch-in-der-lieferkette-v1.0-2019-12

[9] "IDTA 02068-1-0 Submodel Template - Provision of CompanyData," Industrial Digital Twin Association, Germany, Tech. Rep., 2025, under review.

[10] "IDTA 02002-1-0: Submodel Template - Contact Information," Industrial Digital Twin Association, Germany, Tech. Rep., 2022. [Online]. Available: https://github.com/admin-shell-io/submodel-templates/tree/main/published/Contact%20Information/1

[11] "IDTA 02004-1-2: Submodel Template - Handover Documentation," Industrial Digital Twin Association, Germany, Tech. Rep., 2023. [Online]. Available: https://github.com/admin-shell-io/submodel-templates/tree/main/published/Handover%20Documentation/1/2

[12] "IDTA 02003-1-2: Submodel Template - Generic Frame for Technical Data for Industrial Equipment in Manufacturing," Industrial Digital Twin Association, Germany, Tech. Rep., 2022. [Online]. Available: https://github.com/admin-shell-io/submodel-templates/tree/main/published/Technical_Data/1/2

[13] "IDTA 02006-3-0: Submodel Template - Digital Nameplate for Industrial Equipment," Industrial Digital Twin Association, Germany, Tech. Rep., 2024. [Online]. Available: https://github.com/admin-shell-io/submodel-templates/tree/main/published/Digital%20nameplate/3/0

[14] "Asset Administration Shell Specification Part 4: Security," Industrial Digital Twin Association, Germany, Tech. Rep., 2024. [Online]. Available: https://industrialdigitaltwin.io/aas-specifications/IDTA-01004/v3.0/index.html

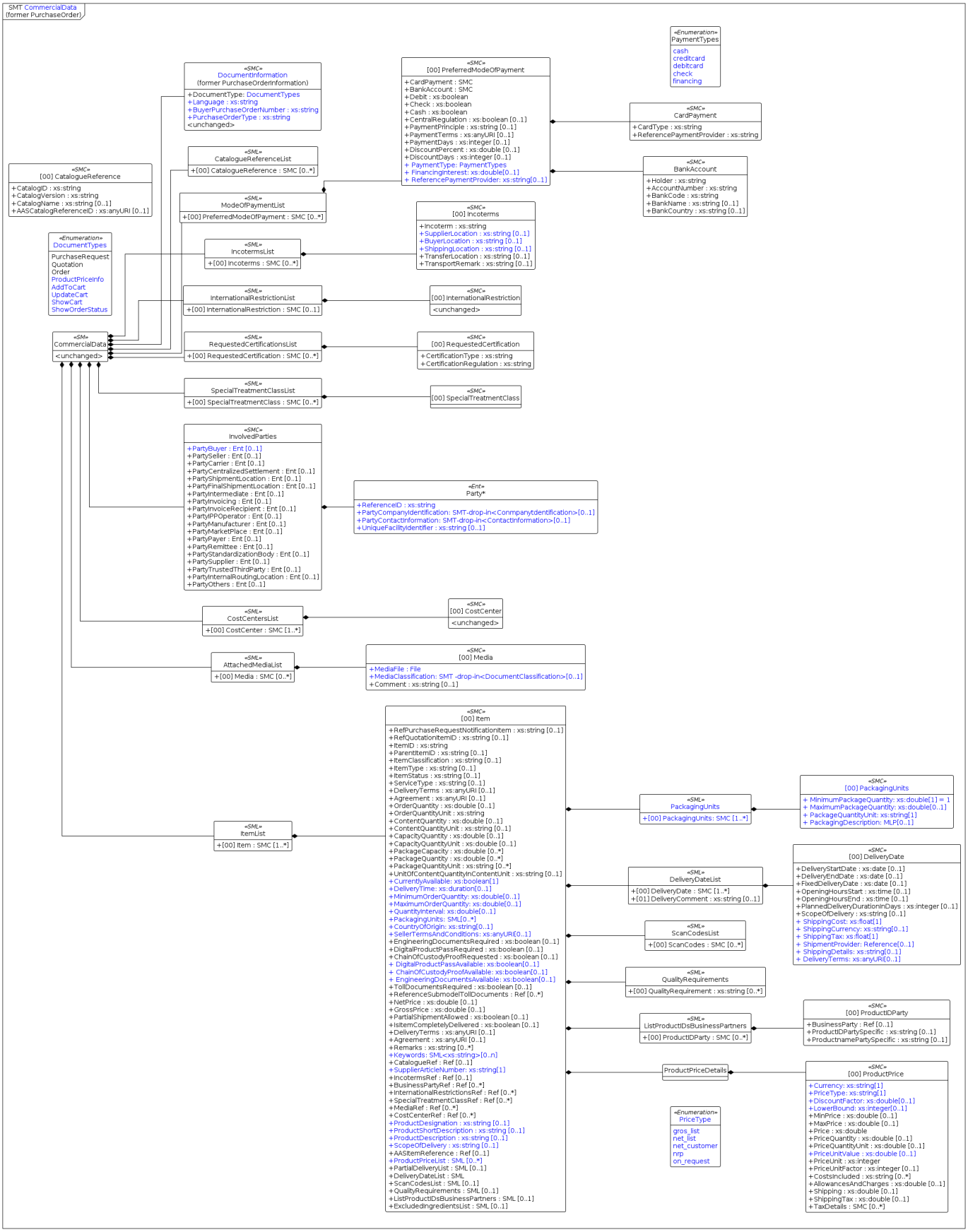


Fig. 4: Submodel PurchaseOrder extended and renamed to CommercialData