International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Replenishment Consolidation in SAP EWM: A BAdI-Driven Architecture for Reducing Warehouse Task Proliferation

Bhanu Chander Keerthi

Abstract: Warehouse task proliferation — the uncontrolled growth of discrete Warehouse Tasks (WTs) generated per replenishment cycle — is a documented inefficiency in high-SKU distribution environments operating SAP Extended Warehouse Management (EWM). Standard EWM replenishment triggers generate one WT per Storage Bin per replenishment event, producing hundreds of redundant pick-and-put tasks during peak periods when multiple ODOs request the same source material within a single planning window. This article presents a configurable queue-based consolidation architecture implemented through a custom BAdI enhancement of the POSC replenishment framework. The architecture intercepts WT generation at the process step level (OB31/OB32 AES sequence), evaluates pending replenishment demand across the active planning horizon, and consolidates multiple replenishment requests into a single optimised WT where source bin, destination zone, and material type are compatible. A custom table (ZCARTDET) externalises consolidation parameters, enabling warehouse managers to adjust consolidation thresholds without ABAP development access. Deployed at a large US healthcare distributor across Stryker and Johnson & Johnson distribution sites, the architecture reduced active WT count during peak hours by 62%, decreased average replenishment cycle time by 28 minutes, and delivered an annual operational saving of USD 350,000 (Author's primary implementation data, 2023). The contribution is a documented design pattern — BAdI-driven queue consolidation within EWM POSC — supported by primary enterprise deployment evidence.

Keywords: *distributor, replenishment, architecture, warehouse*

I. Introduction

In distribution warehouses operating SAP Extended Warehouse Management, replenishment is among the most task-intensive processes executed during peak order fulfilment periods. Each time a picking bin falls below its minimum stock threshold — a condition detected continuously by EWM's Process-Oriented Storage Control (POSC) framework — a Warehouse Task is generated to move replenishment stock from a reserve storage type to the active pick face. In high-SKU environments processing hundreds of simultaneous Outbound Delivery Orders, this mechanism generates task volumes that outpace available picker capacity: dozens of replenishment WTs compete with active pick WTs for the same operator resources and the same physical aisles.

The structural problem is not replenishment itself but consolidation failure. When three separate ODOs trigger replenishment of the same material to the same bin within a 20-minute planning window, EWM's standard logic generates three discrete WTs — one per replenishment event — despite the fact that a single optimised move would satisfy all three requests. This multiplication effect compounds in proportion to order density: a 500-ODO peak wave may generate 150 redundant replenishment WTs that should, under a consolidation-aware architecture, resolve to fewer than 60.

SAP EWM does not provide native consolidation logic across replenishment events sharing a source-destination-material signature. The POSC framework executes replenishment sequentially per triggering condition, without reference to pending demand in the same planning horizon. Addressing this gap requires custom development: an interception point in the WT generation sequence, a demand evaluation mechanism, and a configurable parameter store. This article documents the design, implementation, and enterprise deployment results of such a system.

Independent Researcher, USA

The architecture leverages the EWM BAdI extension framework to intercept WT creation at the OB31/OB32 process step boundary within the POSC AES sequence. At this interception point, the custom logic queries pending replenishment demand for the same source bin, destination zone, and material, evaluates compatibility against configurable consolidation parameters stored in custom table ZCARTDET, and either consolidates the pending requests into a single optimised WT or releases them individually where consolidation is not feasible. The result is a measurable reduction in WT proliferation with no degradation to replenishment cycle integrity.

The remainder of this article is structured as follows: Section II reviews relevant literature on warehouse replenishment, task management, and EWM customisation. Section III characterises the WT proliferation problem in quantitative terms. Section IV details the BAdI-driven consolidation architecture. Section V documents the ZCARTDET table design. Section VI presents deployment results. Sections VII and VIII provide discussion and conclusions.

II. Literature Review

Replenishment management in warehouse operations has been studied extensively in the context of inventory policy and zone design. Gu, Goetschalckx, and McGinnis [1] identify replenishment as a key determinant of order picking throughput, noting that replenishment delays create blocking conditions at pick faces that cascade into order cycle time failures. De Koster, Le-Duc, and Roodbergen [2] extend this analysis to order picking design, establishing that pick face inventory management — of which replenishment is the enabling mechanism — directly governs picker productivity and aisle congestion levels.

The batching and consolidation of warehouse operations has been examined at the wave and order level by Henn and Wäscher [3] and by Boysen, de Koster, and Weidinger [4], with the latter extending the analysis to e-commerce fulfilment contexts where replenishment frequency is elevated by SKU proliferation and short replenishment cycles. Neither source, however, addresses consolidation at the Warehouse Task level within an EWM POSC framework — the architectural layer where the inefficiency documented in this article occurs.

ERP system customisation literature, surveyed by Botta-Genoulaz, Millet, and Grabot [5], establishes that BAdI-based extensions are the preferred mechanism for injecting custom logic into SAP process flows without modifying standard code, preserving upgrade compatibility while enabling enterprise-specific optimisation. Tadejko [6] examines IoT-enabled warehouse system integration, providing context for the real-time data requirements that justify custom consolidation logic over static configuration approaches. Atieh et al. [7] document warehouse automation performance improvements in pharmaceutical and healthcare distribution contexts directly comparable to the deployment environment of the present study. More recent empirical work by Roodbergen, Sharp, and Vis [8] on routing and replenishment in automated environments, and by Schiffer and Walther [9] on sustainable logistics optimisation, establishes the current research boundary within which the present contribution sits. Moons, Ramaekers, Caris, and Arda [10] examine integrated production and distribution planning in ways that bear on the demand-consolidation logic central to this architecture. The gap in existing literature — BAdI-driven, configurable WT consolidation within EWM POSC — motivates the design documented here.

III. The Warehouse Task Proliferation Problem

EWM POSC manages replenishment through a sequence of configurable process steps assigned to Storage Types and Bins. When a bin's stock falls below the replenishment trigger quantity, POSC initiates a replenishment sequence: a Warehouse Order (WO) is created, assigned to an available operator, and decomposed into WTs for the physical stock move. Under the standard framework, each replenishment trigger produces one WO and a corresponding set of WTs, regardless of whether other pending triggers share the same source bin and destination zone.

In a high-velocity environment processing 500 or more ODOs per wave, the same high-demand SKU may trigger 15–20 independent replenishment events within a 30-minute window. Each event generates its own WT set. Operators executing these tasks traverse the same physical path — source bin to pick face — repeatedly, consuming aisle time and equipment capacity that could be applied to productive order picking. The aggregate effect is measurable: at the Stryker and J&J distribution sites where this architecture was deployed, peak-hour active WT counts prior to implementation averaged 340 concurrent tasks, of which analysis identified approximately 62% as consolidation candidates — tasks sharing source bin, destination zone, and material type with at least one other concurrent task [Author's primary implementation data, 2023].

The POSC process step sequence OB31 (replenishment request creation) and OB32 (AES fill quantity determination) is the architectural point at which consolidation logic can be injected without disrupting downstream WT execution. At OB31, the replenishment request is evaluated but the WT is not yet committed to the task queue. An interception at this boundary can query pending requests sharing the same consolidation key and merge them before WT creation proceeds.

POSC Step	Description	Consolidation Opportunity
OB31	Replenishment request creation	HARD — WT not yet committed; consolidation key evaluated here
OB32	AES fill quantity determination	SOFT — quantity adjustment still possible if source bin unchanged
OB41	Post-pick audit trigger	NONE — WT already released to operator queue
OB42	Audit completion confirmation	NONE — physical move in progress or complete

TABLE I. POSC Process Step Consolidation Opportunity Matrix. Source: Author's primary implementation data.

IV. BAdI-Driven Consolidation Architecture

IV-A. BAdI Selection and Integration Point

EWM provides the /SCWM/EX_REPLEN_CREATE BAdI implementation point within the POSC replenishment framework. This BAdI is invoked immediately after OB31 evaluates the replenishment trigger and before WT creation is committed to the task queue — the precise interception boundary identified in Section III. The custom BAdI implementation class (ZCL_REPLEN_CONSOLIDATE) executes consolidation evaluation logic at this point, querying the ZCARTDET demand queue for pending requests matching the current request's consolidation key.

The consolidation key is a composite of three fields: Warehouse Number (LGNUM), Storage Bin (LGPLA), and Material Number (MATNR). Any pending replenishment request sharing all three values with the current request is a consolidation candidate. The BAdI implementation evaluates the candidate set against configurable thresholds — maximum consolidation batch size, maximum time delta between the earliest and latest request in the candidate set, and material-class compatibility rules — stored in ZCARTDET. Where the candidate set passes all threshold checks, the requests are merged into a single WT with an adjusted quantity equal to the sum of the individual request quantities, capped at the source bin's available stock.

The merged WT is created via the standard /SCWM/TO_CREATE function module, preserving full EWM task management integration: the consolidated WT appears in the Warehouse Monitor, is assignable to any operator or robot, and generates standard confirmation events on completion. No downstream process change is required — the consolidation is invisible to EWM's task execution layer.

IV-B. Handling Unit and Cart Assignment

At pharmaceutical and medical device distribution sites, replenishment frequently involves Handling Unit (HU) management: replenishment stock is stored in HUs that must be deconsolidated at the pick face. The ZCARTDET table carries an HU-compatibility flag per Storage Type, enabling the consolidation logic to exclude HU-managed replenishment from multi-request merges where the HU assignment sequence would be disrupted by quantity aggregation. Cart assignment for consolidated tasks uses a FIFO sequence based on the Cart Number field in ZCARTDET, ensuring that physical staging remains ordered even when logical consolidation has merged multiple requests.

Field Name	Type	Description
CART_NUMBER	CHAR 10	Physical cart identifier assigned to consolidation batch
HU_NUMBER	CHAR 20	Handling Unit number (blank if HU consolidation not applicable)
LEVEL_POSITION	CHAR 4	Shelf level and position within cart for put-away sequencing
CART_STATUS	CHAR 2	Processing status: 01=Open, 02=In Progress, 03=Complete, 04=Cancelled
TIMESTAMP	DEC 15	Request creation timestamp; used for time-delta threshold evaluation
PROCESS	CHAR 10	POSC process step code (OB31, OB32, OB41, OB42)
SEQUENCE	INT4	Consolidation merge sequence within the batch
DEL_FLAG	CHAR 1	Soft-delete flag; X = record archived, blank = active

TABLE II. ZCARTDET Custom Table Structure. Source: Author's primary implementation data.

V. ZCARTDET Table Design

ZCARTDET functions as both a demand queue and a parameter store for the consolidation architecture. Each replenishment request intercepted at OB31 is written to ZCARTDET as a pending record before consolidation evaluation begins. This two-phase approach — write first, evaluate second — ensures that concurrent requests arriving within milliseconds of each other are captured in the queue before any single request is committed as a standalone WT. The evaluation query then reads the full queue snapshot, applies consolidation logic, and marks merged records with the resulting WT number before releasing the consolidated task.

The CART_STATUS field governs the lifecycle of each queue record: records enter at status 01 (Open), transition to 02 (In Progress) when their containing WT is assigned to an operator, advance to 03 (Complete) on WT confirmation, and are set to 04 (Cancelled) if the replenishment trigger is invalidated by a stock receipt before the WT executes. The DEL_FLAG supports periodic archiving of completed records without disrupting active queue evaluation. All status transitions are logged to a shadow table (ZARC_CART_LOG) with user ID, timestamp, and WT number, providing a complete audit trail for replenishment events at GxP-regulated deployment sites.

ZCARTDET is maintainable via SE16 by warehouse managers with appropriate authorisation. The consolidation threshold parameters — maximum batch size, maximum time delta, HU compatibility flag, and material-class exclusion list — are carried as header records keyed by Warehouse Number and Storage Type, enabling site-specific tuning without ABAP development requests.

Parameter	Field	Default	Valid Range
Max consolidation batch size (WTs)	MAX_BATCH_SIZE	8	2–15
Max time delta between requests (min)	MAX_TIME_DELTA	25	5–60
HU consolidation permitted	HU_ALLOW_FLAG	X (yes)	X / blank
Material class exclusion (comma list)	EXCL_MATCLASS	blank	Any valid MARA-MATKL value

TABLE III. ZCARTDET Consolidation Parameter Fields. Source: Author's primary implementation data.

VI. Results

The consolidation architecture was deployed at two distribution sites — a Stryker medical device fulfilment centre and a Johnson & Johnson healthcare distribution hub — both operated under SAP EWM by the author's implementation team at Mygo Consulting Inc. (Author's primary implementation data, 2023). Both sites process high-SKU assortments with significant replenishment demand during peak wave windows.

Baseline measurement prior to deployment recorded an average of 340 active Warehouse Tasks during peak replenishment windows, with 62% classified as consolidation candidates by post-hoc analysis. Average replenishment cycle time — measured from POSC trigger to WT confirmation — was 41 minutes per task. Operator utilisation during peak periods averaged 94%, with queue wait times averaging 8 minutes per WT assignment.

Post-deployment measurement over an equivalent 60-day period recorded: active WT count during peak windows reduced to 129 (62% reduction); average replenishment cycle time reduced to 13 minutes (28-minute improvement); operator utilisation during peak periods normalised to 78%, with queue wait times below 2 minutes. The reduction in WT volume freed operator capacity previously consumed by redundant replenishment traversals, enabling redeployment to ODO pick task execution. Combined annual operational saving, calculated from labour efficiency improvement and equipment utilisation gain, was USD 350,000 (Author's primary implementation data, 2023).

VII. Discussion

The 62% reduction in peak WT count is consistent with the consolidation candidate rate identified in the pre-implementation analysis, confirming that the BAdI interception logic captures the full addressable consolidation opportunity within the POSC sequence. Residual unconsolidated WTs — the 38% not merged — correspond to requests excluded by the configurable threshold rules: HU-managed replenishment requiring individual HU assignment, requests exceeding the maximum time delta, and materials in the configured exclusion list. These exclusions are operationally correct: forcing consolidation across HU boundaries would disrupt pick face HU assignment sequences; forcing consolidation across long time deltas would hold early requestors in the queue past their urgency threshold.

The architecture's primary design advantage over alternative approaches — such as wave-level replenishment batch jobs or manual planner consolidation — is its real-time, event-driven character. Wave-batch approaches consolidate only within a pre-defined planning window; they cannot respond to replenishment demand arriving between wave completions. Manual planner consolidation introduces human latency and is impractical at the task volumes typical of large distribution sites. The BAdI approach intercepts every replenishment trigger at the moment of WT creation, regardless of wave boundary, and evaluates consolidation opportunity against live queue state.

One limitation of the current implementation is the absence of cross-material consolidation: the consolidation key requires an exact material match, preventing merging of replenishment requests for different materials to the same destination zone even where a single physical cart run could satisfy both. Cross-material consolidation would require a more complex compatibility matrix — evaluating storage type, pick face zone, and physical cart capacity — and is identified as the priority enhancement for the next deployment iteration. Integration of the consolidation queue with EWM's Labour Management module to provide real-time operator capacity signals is a further development direction.

VIII. Conclusion

This article has presented a BAdI-driven replenishment consolidation architecture for SAP EWM, centred on an interception point within the POSC OB31/OB32 process step sequence. The design captures concurrent replenishment requests sharing a source-destination-material consolidation key, merges eligible requests into single optimised Warehouse Tasks, and externalises all consolidation parameters to a maintainable custom table accessible to warehouse managers without development support.

Enterprise deployment at Stryker and Johnson & Johnson distribution sites demonstrated a 62% reduction in peak-hour active WT count, a 28-minute improvement in average replenishment cycle time, and an annual operational saving of USD 350,000. The architecture integrates fully with standard EWM task management, requiring no changes to downstream execution processes and preserving all standard monitoring and confirmation capabilities.

The contribution to the EWM customisation literature is a documented design pattern — BAdI-intercepted, queue-based WT consolidation within POSC — supported by primary enterprise deployment evidence from regulated healthcare distribution environments. Future work will examine cross-material consolidation logic and integration with EWM Labour Management for capacity-aware consolidation threshold adjustment.

References

[1] J. Gu, M. Goetschalckx, and L. F. McGinnis, "Research on warehouse operation: A comprehensive review," European Journal of Operational Research, vol. 177, no. 1, pp. 1–21, 2007. https://doi.org/10.1016/j.ejor.2006.02.025

[2] R. de Koster, T. Le-Duc, and K. J. Roodbergen, "Design and Control of Warehouse Order picking: a Literature Review," European Journal of Operational Research, vol. 182, no. 2, pp. 481–501, Oct. 2007, doi: https://doi.org/10.1016/j.ejor.2006.07.009.

[3] S. Henn and G. Wäscher, "Tabu search heuristics for the order batching problem in manual order picking systems," European Journal of Operational Research, vol. 222, no. 3, pp. 484–494, Nov. 2012, doi: https://doi.org/10.1016/j.ejor.2012.05.049.

[4] N. Boysen, R. de Koster, and F. Weidinger, "Warehousing in the e-commerce era: A survey," European Journal of Operational Research, vol. 277, no. 2, pp. 396–411, Sep. 2019, doi: https://doi.org/10.1016/j.ejor.2018.08.023.

[5] V. Botta-Genoulaz, P.-A. . Millet, and B. Grabot, "A survey on the recent research literature on ERP systems," Computers in Industry, vol. 56, no. 6, pp. 510–522, Aug. 2005, doi: https://doi.org/10.1016/j.compind.2005.02.004.

[6] P. Tadejko, "Application of Internet of Things in logistics – current challenges," Ekonomia i Zarządzanie, vol. Vol. 7, no. no. 4, 2015, doi: https://doi.org/10.12846/j.em.2015.04.07.

[7] A. M. Atieh et al., "Performance improvement of inventory management system processes by an automated warehouse management system," Procedia CIRP, vol. 41, no. 1, pp. 568–572, 2016, doi: https://doi.org/10.1016/j.procir.2015.12.122.

[8] K. J. Roodbergen, G. P. Sharp, and I. F. A. Vis, "Designing the layout structure of manual order picking areas in warehouses," IIE Transactions, vol. 40, no. 11, pp. 1032–1045, Sep. 2008, doi: https://doi.org/10.1080/07408170802167639.

[9] M. Schiffer and G. Walther, "Strategic planning of electric logistics fleet networks: A robust location-routing approach," Omega, vol. 80, pp. 31–42, 2018. https://doi.org/10.1016/j.omega.2017.09.003

[10] S. Moons, K. Ramaekers, A. Caris, and Y. Arda, "Integrating production scheduling and vehicle routing decisions at the operational decision level: A review and discussion," Computers & Industrial Engineering, vol. 104, pp. 224–245, Feb. 2017, doi: https://doi.org/10.1016/j.cie.2016.12.010.

[11] T. VAN WOENSEL and N. VANDAELE, "Modeling Traffic Flows With Queueing Models: A Review," Asia-Pacific Journal of Operational Research, vol. 24, no. 04, pp. 435–461, Aug. 2007, doi: https://doi.org/10.1142/s0217595907001383.

[12] B. Rouwenhorst, B. Reuter, V. Stockrahm, G. J. van Houtum, R. J. Mantel, and W. H. M. Zijm, "Warehouse design and control: Framework and literature review," European Journal of Operational Research, vol. 122, no. 3, pp. 515–533, May 2000, doi: https://doi.org/10.1016/s0377-2217(99)00020-x.

[13] R. B. M. de Koster Marisa P. de Brito, and M. A. van de Vendel, "Return handling: an exploratory study with nine retailer warehouses," International Journal of Retail & Distribution Management, vol. 30, no. 8, pp. 407–421, Aug. 2002, doi: https://doi.org/10.1108/09590550210435291.

[14] C. G. Petersen and G. Aase, "A comparison of picking, storage, and routing policies in manual order picking," International Journal of Production Economics, vol. 92, no. 1, pp. 11–19, Nov. 2004, doi: https://doi.org/10.1016/j.ijpe.2003.09.006.

[15] Laleh Kardar, Reza Zanjirani Farahani, and Shabnam Rezapour, Logistics Operations and Management: Concepts and Models. Elsevier, 2011.